UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA
Departamento de Sistemas Informáticos y Computación

TESIS DOCTORAL

Más sobre equivalencias lógicas y distancias entre procesos

Revisiting logical semantics for processes and their distances

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

David Romero Hernández

Director

David de Frutos Escrig

Madrid, 2016

UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA | DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

DAVID ROMERO HERNÁNDEZ

MÁS SOBRE EQUIVALENCIAS LÓGICAS Y DISTANCIAS ENTRE PROCESOS

Revisiting logical semantics for processes and their distances

TESIS DOCTORAL | 2016 | Director DAVID DE FRUTOS ESCRIG

Más sobre equivalencias lógicas y distancias entre procesos

UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE INFORMÁTICA
Departamento de Sistemas Informáticos y Computación

Memoria para optar al grado de Doctor
presentada por

David Romero Hernández

Dirigida por el profesor
David de Frutos Escrig

Programa de Doctorado en Ingeniería Informática
Enero de 2016

Revisiting logical semantics for processes and their distances

UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE INFORMÁTICA
Departamento de Sistemas Informáticos y Computación

Requirements for the Ph.D. degree
presented by

David Romero Hernández

Supervised by Professor
David de Frutos Escrig

Programa de Doctorado en Ingeniería Informática
January, 2016

A Patricia, Clara, nuestro pequeño ángel del cielo, y aquell@s que (D.m.) están por venir

Más sobre equivalencias lógicas y distancias entre procesos

David Romero Hernández
Facultad de Informática
Universidad Complutense de Madrid

Cubierta: Carlos Romero Hernández

Madrid, enero de 2016

Resumen

Esta tesis se enmarca en el amplio campo de la teoría de la concurrencia. Más específicamente, nos centramos en el estudio de las relaciones de similitud entre procesos concurrentes. Comenzamos estudiando la bisimulación, considerada la más importante de estas relaciones, y vemos después cómo podemos extender nuestros resultados al resto de las semánticas de procesos estudiadas durante las últimas décadas.

En particular, nuestra contribución a la comunidad científica, se centra en dos puntos principales:

– El desarrollo de una caracterización lógica uniforme de las semánticas de procesos: proponemos un esquema lógico común (enmarcado en la conocida lógica modal de *Hennessy-Milner*) e incluimos las diferentes semánticas en este esquema, enfatizando las diferencias y similitudes entre ellas, que se presentan del modo más claro posible.

– La presentación de una nueva noción de distancia, tanto entre procesos finitos como infinitos: la misma se diferencia de las anteriormente propuestas en su carácter global, que acumula las diferencias que aportan los distintos cómputos, en lugar de quedarnos con la máxima de ellas.

La primera parte de nuestra investigación continúa la tarea comenzada por el Prof. Carlos Gregorio Rodríguez y mi propio director, el Prof. David de Frutos Escrig. Las distintas nociones de semánticas de procesos fueron recopiladas en los años noventa por Rob J. van Glabbeek, quien estableció las relaciones existentes entre ellas en su famoso *linear time-branching time spectrum*. C. Gregorio y D. de Frutos pusieron estas semánticas bajo un marco común, clarificando dichas relaciones. Su trabajo dotó al *spectrum* de R. J. van Glabbeek de una estructura mucho más ordenada, y permitió tener una visión unificada de las características de dichas semánticas. Nuestra caracterización lógica completa este trabajo unificador y nos permite descubrir nuevas semánticas desconocidas hasta la fecha. Además, explotando la dualidad existente entre nuestro marco lógico y el marco observacional (desarrollado por mis compañeros), se ponen aún más de manifiesto las características definidoras de cada una de las semánticas.

Una vez que completamos el estudio de las relaciones entre las diferentes semánticas de procesos, apareció de forma natural la siguiente cuestión: *¿Qué*

ocurre con los procesos que no son equivalentes con respecto a la semántica considerada?. Distintas nociones de distancia entre procesos han sido estudiadas por diferentes autores a lo largo de los últimos años. Entre ellas destacan los estudios dirigidos por Thomas A. Henzinger y Ulrich Fahrenberg, que se apoyan en diversas variantes del llamado *juego de bisimulación cuantitativo*. Esta noción de distancia sólo tiene en cuenta la máxima diferencia entre los procesos y no todas ellas en su conjunto. Por ello, nosotros proponemos aquí una nueva noción de distancia, con la que somos capaces de realizar una medida más precisa de las diferencias entre procesos con respecto a las distintas semánticas del *spectrum*. Además podemos extenderla para trabajar con procesos infinitos, utilizando la coinducción como recurso básico para comparar los comportamientos infinitos.

Abstract

This thesis can be included in the broad field of concurrency theory. More specifically, we focus on the study of the similarities between concurrent processes. We start from bisimulation, the main of these relations, and then we see how we can extend the obtained results to the rest of the semantics developed along the last years.

In particular, our main contributions can be roughly described by the following two items:

- The development of a unified logical characterization of process semantics: we propose a common logical scheme (within the framework of the well known *Hennessy-Milner Logic*) and we set the different semantics in this scheme by emphasizing, in the clearest possible way, the (dis)similarities between them.

- We present a new notion of distance for both finite and infinite processes. This novel notion differs from the previously available ones in its global character: instead of taking the maximum disagreement between the two compared processes, it adds all the differences provided by their whole sets of computations.

The first part of our research continues with the work started by Prof. Carlos Gregorio Rodríguez and my supervisor, Prof. David de Frutos Escrig. The different process semantics developed along the lasts decades were collected by Rob J. van Glabbeek in the nineties. He established their relationships in his very well known *linear time-branching time spectrum*. C. Gregorio and D. de Frutos consolidated this *spectrum* providing a nice common framework, where the semantics are classified in a much more systematic way. They endowed the Rob J. van Glabbeek's spectrum with an ordered structure, giving also a unified vision of the different semantics developed in the literature. Our logical characterization completes this unification work and also enables us to find out some new semantics that were not considered before. Furthermore, the duality between our logical framework and the observational one (developed by my colleagues) clarifies the distinctive features of the different semantics.

Once we understood the relationships between the different process semantics, the following question naturally appeared: *What happens when two processes are not equivalent under a given semantics?* Several notions of distance between

processes have been studied by different authors along the last years. We will highlight here the studies guided by Thomas A. Henzinger and Ulrich Fahrenberg. Their notions of distance rely in the so called *quantitative bisimulation game*, taking into account the maximum disagreement between processes, and not all of them at the same time. In our opinion, it is more adequate to consider the sum of all these differences, and hence we have proposed a new framework. Within this framework, we are able to define a more precise notion of distance between processes wrt any of the semantics in the *spectrum*. Finally, we extend this notion to the field of infinite processes, using in this case coinduction as the basic resource to deal with the infinite behaviors.

Agradecimientos

Entre los pecados mayores que los hombres cometen, aunque algunos dicen que es la soberbia, yo digo que es el desagradecimiento, ateniéndome a lo que suele decirse: que de los desagradecidos está lleno el infierno.

El ingenioso hidalgo don Quijote de la Mancha
Capítulo 58, parte 2. Miguel de Cervantes

Sin duda alguna la parte más gratificante de escribir una tesis, al menos en mi caso, corresponde a los agradecimientos. Por un lado en ellos se pueden dejar de lado los rigores científicos permitiendo mostrar el reconocimiento a todas aquellas personas sin las cuales este proyecto no hubiera salido adelante. Por otro lado, y de nuevo al menos en mi caso, a pesar de aparecer en primer lugar, suponen la culminación de esta tesis que con tanto esfuerzo he realizado.

Quisiera empezar por el núcleo principal de mi vida: mi mujer Patricia y mi hija Clara. A Patricia la conocí antes incluso de iniciar mi carrera, con ella he compartido momentos de alegrías y tristezas, fue ella la que me enseñó que todo esfuerzo tiene su recompensa. Desde hace tres años decidimos casarnos y comenzar un nuevo proyecto de vida juntos. En estos últimos años ha sido lo que más echaba en falta cada vez que de nuevo debía ausentarme por un largo periodo. Patricia ha "sufrido" el compartirme con este proyecto que por fin hoy concluye, siendo especialmente difíciles los momentos de separación durante mis estancias en la Universidad de Reykjavík que la dejaban sola al frente de nuestro hogar y al cuidado de nuestra hija. A pesar de todo, siempre ha estado ahí con su alegría y su bondad para ocuparse de nuestra casa, apoyarme en mis decisiones y escuchar mis problemas. Gracias por ser una mujer 10, no cabe duda de que yo no podría haber escrito esta tesis sin tu ayuda. Gracias a mi pequeña Clara que lamentablemente también ha tenido que vivir, a pesar de su corta edad, la ausencia de su padre en los momentos en que mi trabajo ha requerido que pusiese toda mi atención y dedicación sobre él. Ha sido especialmente duro para ambos separarnos durante el tiempo de mis congresos y estancias. Sin embargo, siempre que volvía me recibía con su sonrisa, y una simple mirada de sus ojos azules llenos de vida, bastaba para colmarme de la fuerza y el ánimo necesarios para continuar con la elaboración de mi tesis.

Las líneas son insuficientes para agradecer a mi director David de Frutos su

esfuerzo. Tras finalizar la carrera pensaba que escribir una tesis era una tarea imposible, a pesar de ser algo que siempre quise hacer. Sus palabras fueron claves para animarme a tomar la decisión de intentarlo en lugar de incorporarme al mundo de la empresa. De él he aprendido no solo los conocimientos, sino también la responsabilidad y el rigor académico sin los cuales no podría tener una formación completa como investigador. Gracias por esas discusiones científicas, por tu paciencia, por el tiempo que me has dedicado a pesar de las numerosas tareas que tenías entre manos, y sobretodo gracias porque durante estos años has llegado a ser mucho más que mi director de tesis. Sin tu ayuda este sueño jamás habría podido hacerse realidad.

Volviendo a la familia, tengo que agradecer a mis padres Pedro y Mayte su amor incondicional. A lo largo de mi vida siempre han estado ahí para corregir mis faltas con amor y celebrar mis éxitos con alegría. No cabe duda de que soy lo que soy gracias a ellos y a la educación que me han dado durante estos años. Gracias a mi hermano Carlos que, aunque es difícil, ha sido mucho más que un hermano para mi. Ya desde la infancia hemos tenido una relación muy especial, en él siempre he encontrado palabras de apoyo, ánimo y consuelo, tan necesarias en determinados momentos de mi vida. Considero que en parte este logro también es tu logro, en el que tu participación ha quedado puesta de manifiesto con el diseño de la cubierta de esta tesis. Gracias por ser como eres.

No puedo olvidarme de mi familia política. Mi cuñado José (Chema con cariño, aunque me cueste alguna reprimenda), que se convirtió en una parte importante de mi vida antes incluso de que fuéramos familia. Gracias por las fiestas, viajes, juergas, tardes de *Pro* y momentos de ocio que hemos compartido juntos. Sin todos ellos, el largo tiempo dedicado al trabajo habría sido insoportable. Gracias a mis suegros José Ángel y Rosario que, en especial durante estos últimos años, también han estado a mi lado animándome a la conclusión de este proyecto y cuidando durante mi ausencia de la razón de mi existir: Patricia y Clara.

Ya en la parte de ciencia, quisiera empezar agradeciendo a mis compañeros de departamento (muchos de ellos mis profesores durante la licenciatura o el máster): Yolanda Ortega (por su informe detallado sobre esta tesis), M^a Inés Fernández, Carlos Gregorio, Miguel Palomino, Jesús Escribano, Fernando Rosa, Paco López, Manuel Núñez, Mercedes G. Merayo, Ricardo Peña, Clara Segura y Sonia Estévez, con los que he compartido comidas, charlas, clases, congresos, artículos... y que han hecho más amena mi vida en la universidad. Reservo un hueco especial para Ismael Rodríguez, que amablemente accedió a revisar esta tesis cuando aún estaba en una versión preliminar. Gracias a Paloma López y Lidia Sánchez, mis fabulosas compañeras de despacho, que me han permitido "desconectar" de la ciencia a través de esas entretenidas conversaciones. Ignacio Fábregas siempre estará pre-

sente en mi memoria. Con él he vivido grandes momentos en Reykjavík y, además de ser un gran amigo, también ha contribuido a mejorar la presentación de esta tesis leyéndose la versión final de la misma durante su tiempo libre entre nuestros numerosos planes en la capital islandesa. Sus consejos sobre LaTeX, especialmente durante mis primeros años, me sacaron de más de un aprieto. Gracias a mi compañera de promoción M^a Rosa Martos, que durante mis ausencias me sustituyó en las clases que debía dar, y que a su vez también me regaló su amistad.

Mil gracias a Narciso Martí, no solo por ser quien en mis primeros años como investigador me sustentó económicamente con su proyecto, sino por su ayuda en la conclusión de esta tesis. Su revisión de la misma, sus pequeñas sugerencias en la presentación para adaptarse a la normativa, y su conocimiento a la hora de llevar a cabo todo el papeleo que conlleva la elaboración de una tesis, han sido un apoyo fundamental. Contar con la presencia de una gran persona como es Narciso hace mucho más sencillo todos estos trámites, gracias por no dudar en ningún momento en dedicarme parte de tu tiempo aderezado con el peculiar sentido del humor que te caracteriza.

Quisiera agradecer también a Luca Aceto y Anna Ingólfsdóttir su gran hospitalidad en la Universidad de Reykjavík, en la que he estado durante más de tres meses gracias a la beca *EEA Grants* del proyecto *Formal Methods for the Development and Evaluation of Sustainable Systems*. Allí, hemos comenzado juntos nuevas investigaciones que espero den como resultado final la publicación de nuevos artículos. A Anna también tengo que agradecerle el informe que realizó de esta tesis, que sin duda ha contribuido a mejorar su presentación. Y como la cosa va de informes, agradezco también al tercer informante de mi tesis Mohammad Reza Mousavi sus detallados comentarios sobre la misma y el esfuerzo realizado para entregarlos a tiempo.

Por último quisiera agradecer a mis amigos (perdonad mi torpeza si alguno no aparece en esta lista): José Manuel, Fran, Esther, Inma, M^a Ángeles, Javi, Ana, Ernesto, Patxi, Nieves, Bea, Raúl, Cris, todos esos momentos, rezos, conversaciones... que han servido para que pueda concluir felizmente este trabajo. A todos vosotros simplemente GRACIAS.

Las publicaciones de este trabajo han sido realizadas como parte de los proyectos: *DESAFIOS10: Desarrollo de software de alta calidad, fiable, distribuido y seguro* (TIN2009-14599-C03-01 (Proyecto Coordinado)). *PROMETIDOS: Programa en métodos rigurosos para el desarrollo de software* (S2009-TIC1465). *Beca predoctoral complutense de la UCM* para la formación del personal investigador (convocatoria N^o 14-11-2011). *STRONGSOFT: Tecnologías rigurosas para software de nueva generación abierto* (TIN2012-39391-C04-04 (Proyecto Coordinado)) y *UCM-Santander grant GR3/14*. La beca *EEA Grants* del proyecto *Formal Methods for the Development and Evaluation of Sustainable Systems* (001-ABEL-CM-2013).

Contents

Part I

Contents of the thesis

"... so I say that whoever prints a book exposes himself to great danger, since it is utterly impossible to write in a way that will satisfy and please everyone who reads it."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter-3 Part-2

"... y así, digo que es grandísimo el riesgo a que se pone el que imprime un libro, siendo de toda imposibilidad imposible componerle tal, que satisfaga y contente a todos los que le leyeren."

Chapter

Introduction

This chapter gives an overview of the current status of the field where this thesis is placed, and presents the evolution of this field over the years. We start with a brief history of Computer Science which by no means pretend to be accurate or complete, and in fact could be perfectly avoided if the reader is only interested in the technical contributions of this thesis. At the same time, we decided to include no references in this "light" historical revision.

Because of the two (different but completely connected) lines of research followed in the development of this thesis, we continue by introducing as state of the art, some basic notions on process algebra, process semantics, logical characterizations and distances. To do this, we try, as much as possible, to leave aside their more complex technical issues, and we will gradually immerse into the main ideas that underlie the understanding of this thesis. The chapter ends with a summary about the genesis of this thesis followed with a detailed discussion of our objectives. Finally, we conclude with the structure of the rest of the thesis.

1.1 Computers: from human beings to machines

In the ancient times, "computer" was the name given to those who had the job of performing the repetitive calculations required to compute such necessary things in astronomy, cartography, or military strategies. These boring tasks, apart from the huge number of people needed to perform them, could easily lead workers to carelessness and mistakes. Considering the obvious limitation of doing calculations with the fingers, and afterwards with the help of scribbled sheets, along the centuries the effort to mechanize these tasks became natural and numerous. The *abacus* was the first machine created by the ingenuity of men; this simple object facilitated a person to compute additions and subtractions at the same speed as a person with a hand calculator in the current times (multiplication and division were slower). Although the invention of the abacus is usually assigned to China, the oldest of these simple machines came from Babylonia as far as 300 B.C. They evolved to more and more complex machines, with more sophisticated designs, but always following the same simple ideas.

The 17^{th} century started with a theoretical invention that was critical in the development of practical computation. We are talking about the logarithm, developed by the scotsman John Napier in 1617, which enabled to compute multiplications by reducing them to additions. Next we have two important and more practical milestones: the *calculating machine* designed by Blaise Pascal in 1642, as an aid for his father, who was a tax collector. The *Pascaline* (name given to the Pascal calculating machine) could compute additions and subtractions, and it was the materialization of the drawings by Leonardo da Vinci (1452-1519) of his

gear-driven calculating machines, apparently never built at his time. Just a few years after Pascal, Gottfried W. Leibniz managed to create the *stepped reckoner* (designed in 1671 and built in 1973) which was able to compute the four basic operations. Furthermore, although this machine employed the decimal number system, Leibniz was a strong supporter of the binary system. This system has been essential in the operations of modern computers, since binary digits can be easily represented by on and off states of a switch.

Already in the 19^{th} century, machines were developed evolving the works by Leibniz. The English mathematician Charles Babbage designed in 1832 a never built machine, that he eventually called *The Analytic Engine* (it was an improvement of his previous *Difference Engine*, started in 1822). This new machine would had been able to execute any mathematical operation, even those in tables of numbers, such as logarithm tables. Although, of course, it continued having the restriction of being purely mechanical, *the Analytic Engine* was a great step in the way to our actual computers. Babbage realized that punched cards could be employed as storage mechanisms holding up to 1000 numbers with 40 digits. They could be used for further reference or even as auxiliary functions. The two main parts of his machine were called "Store" (where the number was) and "Mill" (where they evolved to new results) and they led us, respectively, to the *memory unit* and the *central processing unit (CPU)* of modern computers.

The next breakpoint bring us to 20^{th} century, where the electricity and the electronic developments made possible to replace for the first time the engine systems by electric pulses, thus leading to the first computers (exclusively for military uses during the Second World War). *Harvard Mark I* appeared in 1944, operating on 23 digits wide numbers. It had the power of adding or subtracting two of these numbers in three-tenths of a second, and it could store 72 numbers (even though it had three-quarters of a million components!). Forty five years later, computers can perform these operations in a billionth of a second, and they can store 30 million numbers in RAM and another 10 billion numbers on their hard disk. Another candidate to be named granddad of modern computers was *Colossus*, built also during W.W. II by the British for the purpose of breaking the cryptographic codes used by the Germans. Among others mathematicians, Alan Turing was involved in its development. The common forefather of today's computers is *ENIAC*, built between 1943 and 1945 by John Mauchly and J. Presper Eckert. *ENIAC* was the first computer with practical purposes: to replace all the women-computers who calculated the firing tables for the army's artillery guns. Thanks to the removal of moving parts, it ran much faster than the *Mark I*; while the latter required 6 seconds to perform a multiplication, the former only took 2.8 thousandths of a second! The first task assigned to *ENIAC* was to

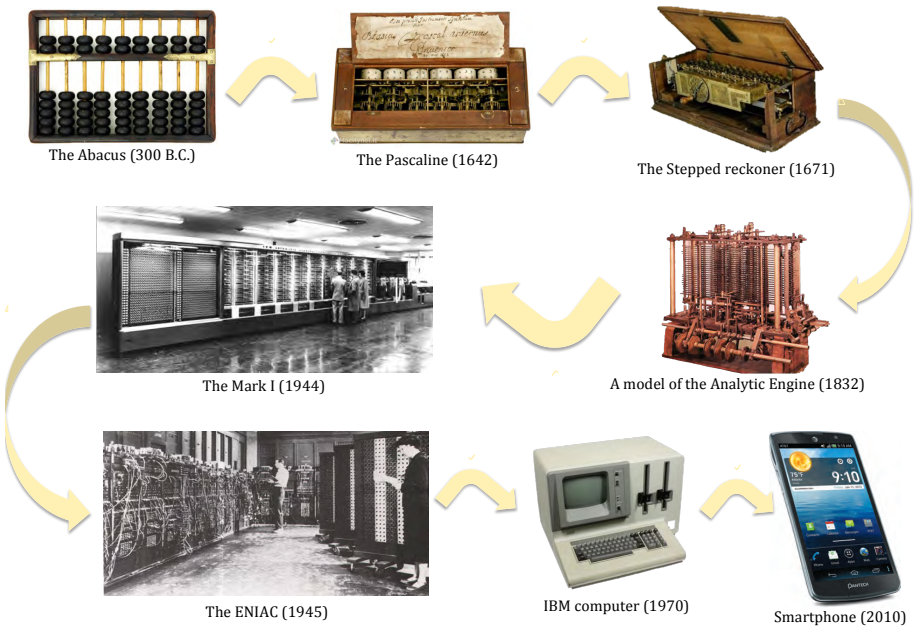


Figure 1.1: Timeline of the computer evolution.

decide whether or not it was possible to build a hydrogen bomb. After chewing half a million punched cards for six weeks, *ENIAC* declared the hydrogen bomb feasible. Later, in collaboration with the mathematician John von Neumann, the two father's of *ENIAC* designed *EDVAC*. It introduced the "comercial" use of the stored program. After these primitive computers others came, such as *ILLIAC*, from University of Illinois, which was the first von Neumann architecture computer built and owned by an American university; *JOHNNIAC* named in this way as a nod to the figure of John von Neumann; and *MANIAC*, based on the von Neumann architecture of the *IAS* machine. In the fifties, *UNIVAC*, a contraction of *Universal Automatic Computer*, appeared as the starting point of the truly commercial computer, accessible to anyone.

After that, *IBM* took the baton having more sales than *UNIVAC* and the computers grew up exponentially. In the middle of the sixties the portability problem was addressed: it would be desirable that a program written for a computer could be executed in any other without any (or at most only a few) change. This fact led to the standardization of programming languages creating among others: FORTRAN (introduced by John Backus [Bac54]), COBOL, ALGOL (studied by Hans Bekic [Bek84]), or even BASIC (*Beginners All-purpose Symbolic Instruction Codes*), that was developed by thinking in those who wanted to become

programmers. As a result structured programming emerged; although it took time to forget about the omnipresent *goto* instruction, even if Corrado Böhm and Giuseppe Jacopini had proved in 1966 that every program could be written just using the three basic control structures: *subroutines, block structures and for and while loops*. Next, in 1968 Edsger Dijkstra wrote his famous letter *Go to statement considered harmful* and this was definitely the starting point of the spreading of structured programming principles that nowadays is still the right way of writing programs. Certainly, we also have to talk here about OOP (*Object-Oriented Programming*), that is a programming paradigm based on the concept of objects, seen as a way of organizing subroutines and data. It is perhaps debatable if this is indeed a new paradigm (as it was presented by their former developers): in our personal opinion it is more a version of structured programming where emphasis is somehow moved from code to data.

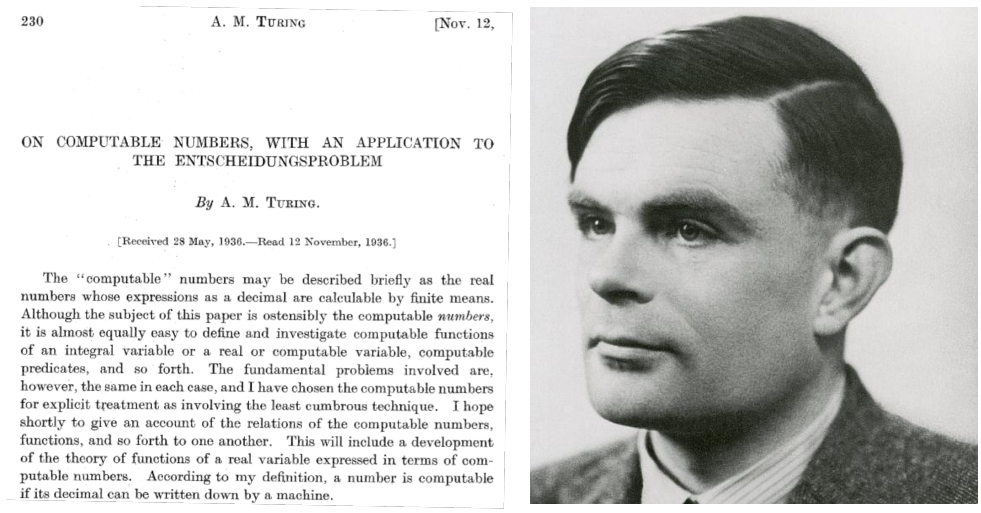
Let us next briefly present the evolution of programming. In the seventies one had to prepare a program off-line on a key punch machine, obtaining the punched cards that were a bright decomposition of (structured) programs into (same size) pieces. Each card held a single statement, and the way to submit your program to the mainframe was to place the stack of cards into the card reader. Next, you had to wait for hours or, even days, till the printout was showed (if everything had worked) hopefully reporting the successful execution of the program. In the eighties we had a slow spread of the personal computers. At the last years of that decade things changed faster, and in the 90's a university student would typically own a computer. But, what phenomenon caused this spin? For sure the invention of the microprocessor, developed by Intel in 1971, contributed to this event. Although Intel did not discover a new computer, they were the first in overstocking an entire computer on a single chip. First Intel's microprocessor: *the Intel 4004*, had 2300 transistors and was clocked at 108 kHZ (i.e., 108,000 times per second), which is almost nothing compared with the 1900 millions of transistors and the 3.7GHz clock rate (i.e. 3,700,000,000 times per second) developed by Intel in his 2015 *Duo-core + GPU Core i7 Broadwell-U*. Moreover, the microprocessors allowed an exponential decrease in the production costs: the second Intel microprocessor, *the 8080*, cost 360 dollars, which was extremely cheap compared to the IBM's famous 360 mainframe, which had a cost of millions of dollars.

These technologies facilitated us to create many different machines that make our lives easier such as GPS, smart phones, smart watches and tablets. To conclude this short review of computer history, let us make a wink to the "history of the future". Let us talk about the summum goal in Computer science: perhaps to create a computer such as *HAL9000*, from the novel *2001: A Space Odyssey* by

Arthur C. Clarke. This sophisticated computer should be able to communicate directly with people, and to self-program performing any task requested to "him". This would mean to create an intelligent creature, even brighter than human (as has been the case in several science fiction films and books; take as a very recent example *Ava* from, the film *Ex Machina*, directed by Alex Garland).

If we finally "triumph", probably we will be happy considering ourselves successful after creating a creature that overpasses human intelligence. But perhaps this will not have the desired consequences (see again the previously cited film for further details). Do you think it will be possible? Can you imagine that? Who knows what tomorrow brings?.

1.2 Process algebra and process semantics



230 A. M. TURING [NOV. 12,

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbrous technique. I hope shortly to give an account of the relations of the computable numbers, functions, and so forth to one another. This will include a development of the theory of functions of a real variable expressed in terms of computable numbers. According to my definition, a number is computable if its decimal can be written down by a machine.

Figure 1.2: Alan Turing (1912-1954).

The previous section gives us an idea of how computers evolved from their origins. The basic principle of our modern computers (controlling the machine's operations by means of a program of coded instructions stored in its memory) was attributed to Alan Turing. In his major publication [Tur36], he introduced what afterwards were called *Turing machines*, consisting of a *tape* (limitless memory), in which (conceptually) both data and instructions are stored (as symbols), and a scanner which is able to move back and forth, reading what it finds and writing symbols (after erasing what it had written). By inserting different programs into its memory, the machine is made to carry out different computations. The afore-

mentioned similar devices, i.e., *Colossus* (Britain 1944) and *ENIAC* (USA 1945), did not yet store programs in their memories. They should be set up for a fresh task by hand, simply modifying some cables connections and setting switches. However, they were based on the (Turing) ideas of operating sequentially, that is, doing only one action (write, read, compute...) at each time.

Over the years, computers became to be more and more sophisticated; in particular, the idea of executing several tasks at the same time appeared. Parallel execution and multitasking arose to implement this concurrent work: while the latter executed each task concurrently, instead of sequentially, but not really at the same time since it used a single CPU; the former was really able to execute at the same time several tasks, taking advantage of the different processors that constitute a modern multicore (or multiprocessors) computer. In this way, a new area emerged that had to cope with these new programs, certainly much more complex than sequential programs. Thus, the *process algebra theory*, as the study of the behavior of parallel (or distributed) systems by algebraic means, appeared.

Jos C. M. Baeten in his paper [Bae05] briefly collects in a bright way the history of process algebra. From its section *What is a process algebra?*, we quote:

> "Process" refers to *behavior* of a *system* [...] "Algebra" denotes that we take an algebraic/axiomatic approach in talking about behavior [...] Likewise, we can say that a process algebra is any mathematical structure satisfying the axioms given for the basic operators. By using the axioms, we can perform calculations with processes. Often, though, process algebra goes beyond the strict bound of universal algebra: we see multiple sorts and binding operators.

Figure 1.3: Left to right: McCarthy (1927-2011), Scott (1932-), Hoare (1934-)

In the early seventies we could distinguish three ways of assigning a meaning

to computer programs. They constitute the process semantics of programming languages:

- *Operational semantics:* that described how a valid program is interpreted as a sequence of computational steps. Properties of a program were verified by constructing proofs from logical statements about its execution and procedures. John McCarthy was a pioneer in this field by using *lambda calculus* to define the semantics in [McC60].

- *Denotational semantics:* whose concern was to find the adequate mathematical object that reflects *what each program exactly does*. These values were taken from the corresponding *semantics domain*, which was chosen depending on the information to be captured in each case. The idea of using continuous domains and the continuous functions between them was first proposed by Dana Scott, who developed the first denotational semantics with Christopher Strachey in [SS71].

- *Axiomatic semantics:* which was based on mathematical logic to prove the correctness of programs. It worked with assertions which are logical statements (or predicates) with variables. These thoughts were developed by Charles Antony Richard (Tony) Hoare in [Hoa69], although he always insisted on having taken his basic ideas both from Robert Floyd [Flo67] and even Alan Turing [CL13].

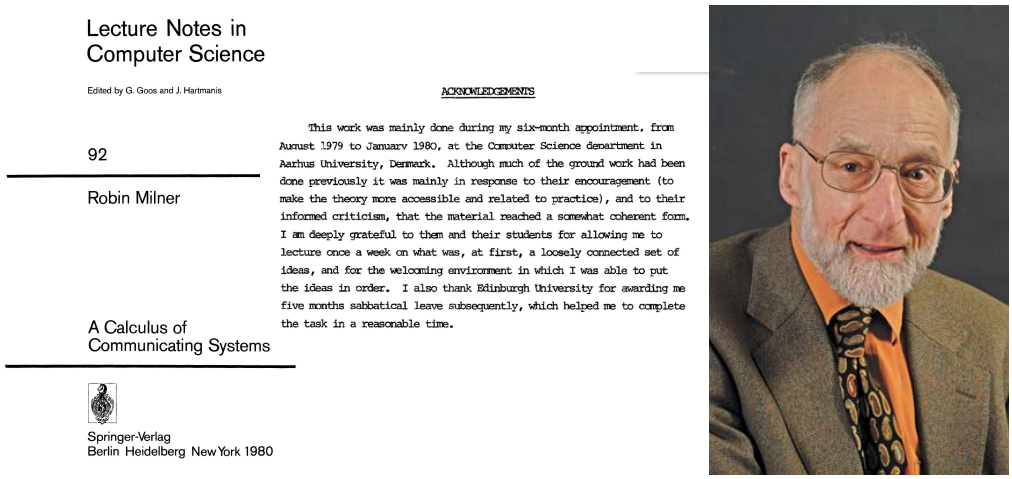
However, those first approaches only covered sequential programming. To assign a meaning to the parallel operator (and therefore to concurrent programs) by using the methods described above was certainly not easy. It was in the last seventies, with the jobs by Gordon D. Plotkin [Plo76] and Susan S. Owicki [Owi76], when the first extensions of different semantics covering also the parallel operator were developed.

Robin Milner was possibly the main person in the history of process algebra. With his book *A Calculus of Communicating Systems* [Mil80], the process algebra reached its first maturity. There, Milner collects his previous work in the process algebra over the years 1973 to 1980: the *Calculus of Communicating Systems (CCS)* within the framework of (abstract) operational semantics included bisimulation as the semantic basis for it.

The syntax of (basic) *CSS* is defined by the following BNF grammar:

$$P ::= \emptyset \mid a.P_1 \mid A \mid P_1 + P_2 \mid P_1|P_2 \mid P_1[b/a] \mid P_1\backslash a$$

As it can be previously seen, starting from the empty process and the prefix operator, the syntax of *CCS* includes primitives for describing choices between


Lecture Notes in
Computer Science

Edited by G. Goos and J. Hartmanis

92

Robin Milner

A Calculus of
Communicating Systems

Springer-Verlag
Berlin Heidelberg New York 1980

ACKNOWLEDGEMENTS

This work was mainly done during my six-month appointment, from August 1979 to January 1980, at the Computer Science department in Aarhus University, Denmark. Although much of the ground work had been done previously it was mainly in response to their encouragement (to make the theory more accessible and related to practice), and to their informed criticism, that the material reached a somewhat coherent form. I am deeply grateful to them and their students for allowing me to lecture once a week on what was, at first, a loosely connected set of ideas, and for the welcoming environment in which I was able to put the ideas in order. I also thank Edinburgh University for awarding me five months sabbatical leave subsequently, which helped me to complete the task in a reasonable time.

Figure 1.4: Robin Milner (1934-2010).

processes, parallel composition, restriction and relabelling. The operational semantics of the expressions that constitute the language, is defined by a labeled transition system (lts) –See Def. 1 in page 15 for further details–, and the notion of bisimulation. The latter was initially introduced in this framework by David Park in [Par81], although several rephrasing of the notion of bisimulation have appeared in many fields of Mathematics and Computers Science, with no apparent direct connections between them (see Davide Sangiorgi study [San09]). It is well known that the (partially naive) use of "bisimulation" in Milner's first book contained some mistakes that were later solved in [Mil89], were he already uses the correct coinductive approach to bisimulation. Milner also formulated his basic CCS in a join work with Matthew Hennessy [HM85] (an updated version of his paper *On Observing Nondeterminism and Concurrency* [HM80]). Furthermore, this paper collects the well known *Hennesy-Milner Logic*, used to specify properties of labeled transition systems.

The eighties were the happy years of process algebra. In parallel and independently to Milner's job, Hoare defined the *Communicating Sequential Processes (CSP)* [Hoa78]. Hoare suggested that the basic primitives of programming were input and output, while parallel composition was a fundamental structuring method. Also, he proposed to distinguish between *internal* and *external nondeterminism*, using two different operators. The syntax of *CSP* defined the way in which processes and events may be combined:

$$\begin{aligned} Proc ::= {} & STOP \mid SKIP \mid e \to P \mid Proc \,\Box\, Proc \mid Proc \sqcap Proc \mid \\ & Proc|||Proc \mid Proc|[\{X\}]|Proc \mid Proc\backslash X \mid Proc; Proc \mid \end{aligned}$$

$$if\ b\ then\ Proc\ else\ Proc \mid Proc \triangleright Proc \mid Proc \triangle Proc$$

Here, besides the events e we have prefixing, external choice, nondeterministic choice, interleaving, interface parallel, hiding, sequential composition, boolean conditional, timeout, and interrupt.

The book *Communicating Sequential Processes* [Hoa85] gave a good overview of CSP, in this case mainly focussing on the denotational semantics framework. The semantics was first described by traces, but immediately Hoare realized that, in order to handle satisfactorily both the (absence of) deadlocks and divergence, more sophisticated domains were needed: thus *the stable failures model* and *the failures/divergence model* were introduced.

In the axiomatic framework, we could perhaps include the *Algebra of Communicating Processes (ACP)* developed since 1982 by Jan A. Bergstra and Jan Willem Klop [BK83, BK84, BK85]. They started by studying the theory of process algebra, that in [BK84] was extended with communication to yield the basis of *ACP*. This theory fundamentally adopted an axiomatic algebraic approach to the formal definition of its operators, starting from the following basic axioms, where $+$ represents *choice or union* and *sequential composition* is simply represented by concatenation of its arguments:

$$x + y = y + x \qquad (x + y) + z = x + (y + z) \qquad x + x = x$$
$$(x + y)z = xz + yz \qquad (xy)z = x(yz)$$

Since *ACP* was devoted to deal with concurrency and interaction, it included the following operators: *merge* $\|$, *left merge* $\|\lfloor$ and *communication* $|$, whose corresponding axioms were:

$$x \parallel y = x\|\lfloor y + y\|\lfloor x + x|y \qquad ax\|\lfloor y = a(x \parallel y) \qquad a\|\lfloor y = ay$$
$$(x + y)\|\lfloor z = (x\|\lfloor z) + (y\|\lfloor z) \qquad ax|b = (a|b)x \qquad a|bx = (a|b)x$$
$$ax \mid by = (a|b)(x \parallel y) \qquad (x + y)|z = (x|z) + (y|z) \qquad x|(y + z) = x|y + x|z$$

We can find a great overview of *ACP* in the book *Process Algebra* [BW90] by J. C. M. Baeten and W. Peter Weijland, and more recently in [BBR10] where the whole work on the subject has been collected in an encyclopedic way.

Therefore, we have included *ACP* as an example of the axiomatic approach, because its authors used axioms to formalize the semantics. However, this axioms directly captured the equivalence of terms and not their operational semantics, as it was the case of *Hoare logics*.

During the last few decades a number of important developments have occurred around the three basic process algebraic theories presented before. Most

of these developments are collected in the magnificent handbook [Ber01]. Next, we will briefly present some of them.

Bisimulation. Strong bisimulation, defined in [Par81], is the central notion of equivalence in process theory. Bisimulation identifies systems that behave in the same way; roughly, one system simulates the other and vice versa. We will see later that the notion of bisimulation will be central in this thesis. *Structural Operational Semantics (SOS)*, introduced by G. D. Plotkin in [Plo81] (which has been reprinted in [Plo04]), was the main way of providing a model to process algebras. The underlying idea consisted of defining the behavior of a system in terms of the behavior of its parts. For this purpose, *SOS* specifications used inference rules defining the valid transitions [AFV01]. There are many technical results based on the *formats* of these rules, as it has been discussed in [Ace03, MRG07].

Time. Timed extensions of process algebras were soon developed in any of the three semantics frameworks mentioned above. George M. Reed and Andrew W. Roscoe in [RR88] were the first that extended *CSP*, while *CCS* with time was introduced by Faron Moller and Chris M. N. Toftsand in [MT90] and Wang Yi in [Yi90]. Meantime the *ACP* time extension started with J. C. M. Baeten and J. A. Bergstra work in [BB91]. Last but not least, Yolanda Ortega Mallén and David de Frutos Escrig studied the axiomatizations of semantics for time processes in [OdF91].

Probabilities and stochastic. Later on, the process algebras were extended with probabilistic or stochastic information, whose characteristics (durations of actions, delays, etc.) are governed by probability distributions. Hans A. Hansson presented in [Han94] one of the first examples of a work in this area. As before, there were probabilistic extensions for each process algebra tradition: in the *CSP* framework we can find the work by Gavin Lowe [Low95]; an example of extension for the *CCS* framework can be found in the work by Jane Hillston [Hil96]; while in the *ACP* tradition we have [BBS95]. The notion of approximation (that naturally led to the notion of distance to which we have contributed with the research presented in this thesis) was particularly important in the area of probabilities. Clearly exact (real) values of the corresponding probabilities are not as important as the fact that those probabilities will be bigger (or smaller) than some concrete bounds. An interesting reference were you can find an unification of several of this models is [HMS08].

There are many other extensions of process algebra not mentioned here, some of them can be found in [Bae05].

1.3 Semantics and logics

In the last decades, the scientific community has proposed several interesting notions of behavioral equivalence depending on the amount of branching structure considered, and on the observations that are enabled at the states of labeled transition systems. Many of these semantics were classified and compared by Rob J. van Glabbeek [vG01] in the so called *linear time-branching-time spectrum (ltbt-spectrum)*. Each semantics can be defined over arbitrary (possibly infinite) processes, whose operational semantics is defined by means of a *Labeled Transition System (lts)*.

***Definition* 1** *A labeled transition system is a tuple* $(Proc, Act, \rightarrow)$ *where Proc is a set of states, Act is a set of labels and* $\rightarrow$ *is a set (of so called labeled transitions) i.e., a subset of* $Proc \times Act \times Proc$. *The fact that* $(p, a, q) \in \rightarrow$ *is written* $p \xrightarrow{a} q$ *and represents that there is a transition from process p to process q with label a.*

If for any given p and a, there exists only a single tuple $(p, a, q) \in \rightarrow$, *then we say that a is* deterministic *for p. If for any given p and a there exists at least one tuple* $(p, a, q) \in \rightarrow$, *then one says that a is* enabled.

Among the semantics studied by R. J. van Glabbeek in [vG01], we can find bisimulation as the most important branching semantics, while between the linear semantics he highlighted: Trace semantics, Readiness semantics and Failures semantics. Let us briefly recall here that linear semantics are those which can be described by linear computations, which corresponds to *linear time* logics such as *Linear Temporal Logic (LTL)* introduced by Amir Pnueli in [Pnu77]. Instead branching semantics take into account the branchings along the full set of computations and thus need to consider the much more complicated *Computational Tree Logic (CTL)*, introduced by Edmund M. Clarke and E. Allen Emerson in [CE81].

Next, we recall the description of these semantics, as R. J. van Glabbeek did in his paper:

> Bisimulation. The concept of bisimulation equivalence stems from Milner [Mil80]. Its formal reformulation is due to Park [Par81]. Milner already proposed a "global testing" framework where processes can be replicated, and thus all the nondeterministic branches of the processes can be tested at the same time. This was later formalized by Samson Abramsky in [Abr89].
>
> Trace semantics. Trace semantics is based on the idea that two processes are to be identified if they allow the same set of observations,

where an observation simply consists of a sequence of actions performed by the process in succession.

Readiness semantics. An observation either results in a trace of the process, or in a pair of a trace and a menu of actions by which the observation could have been extended if the observer wouldn't have blocked them. Such a pair is called a *ready pair* of the process, and the set of all ready pairs of a process is its *ready set*.

Failures semantics. Whenever after a certain sequence of actions σ, the set X of free actions is refused by the process, the set X is called a *refusal set*, and $\langle \sigma, X \rangle$ is a *failure pair*. The set of all failure pairs of a process is called its *failure set*, and constitutes in this case its observable behavior.

Each semantics defines a notion of equivalence between processes; conversely, an equivalence relation (or equality notion) between processes is a simple (although sometimes difficult to understand) way of introducing a notion of meaning for processes. Thus, an equivalence notion tells us when two objects can be considered as different representations of the same conceptual entity. We expect that two processes will be equivalent when they allow the same set of possible observations (in response to certain experiments). Furthermore, the semantics will be partially ordered by the relation *makes at least as many identifications as*, which leads us to the picture of the *ltbt-spectrum* considered by R. J. van Glabbeek in Fig. 1.5.

He studied the semantics in the setting of the process algebra *BCCSP* which only contains the basic process algebraic operators from *CCS (Communication and Concurrency)* and *CSP (Communicating Sequential Processes)*, but is powerful enough to express all the finite non-deterministic processes.

***Definition* 2** *[BCCSP Syntax] Given a set of actions Act which is called the alphabet, the set of processes* BCCSP *is defined according to the following grammar expressed in the* Backus-Naur *form:*

$$p ::= \mathbf{0} \mid ap \mid p + q$$

where $a \in Act$, $\mathbf{0}$ represents the process that does not exhibit any behavior, ap is an application of the unary prefix operator, and $p + q$ is the alternative composition.

The operational semantics of *BCCSP* is quite natural and is based on the following principles:

- $\mathbf{0}$ does not execute any action;
- ap executes the action $a \in Act$ and transforms the original process into the process p;

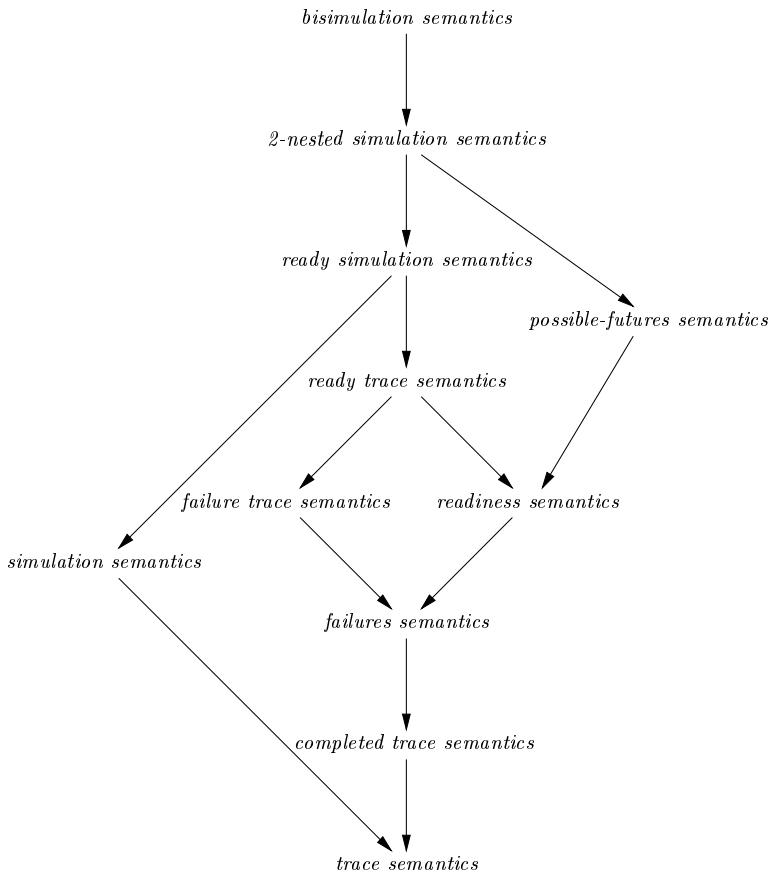


Figure 1.5: Linear time-branching time spectrum as appeared in [vG01].

– $p + q$ is the nondeterministic choice between the behaviors of p and q.

This intuition is captured by the transition rules below, in which a ranges over Act. From these rules we can easily show that the choice operator $+$ is associative and commutative, and has $\mathbf{0}$ as its unit element. Then, finite processes can be represented as finite trees –See the examples in Fig. 1.6–.

Definition 3 *[Operational Rules of BCCSP Semantics] The operational semantics of BCCSP is defined by:*

$$(1)\ \frac{}{ap \xrightarrow{a} p} \qquad (2)\ \frac{p \xrightarrow{a} p'}{p + q \xrightarrow{a} p'} \qquad (3)\ \frac{q \xrightarrow{a} q'}{p + q \xrightarrow{a} q'}$$

Mathew Hennessy and Robin Milner proposed in [HM85] a logical charac-

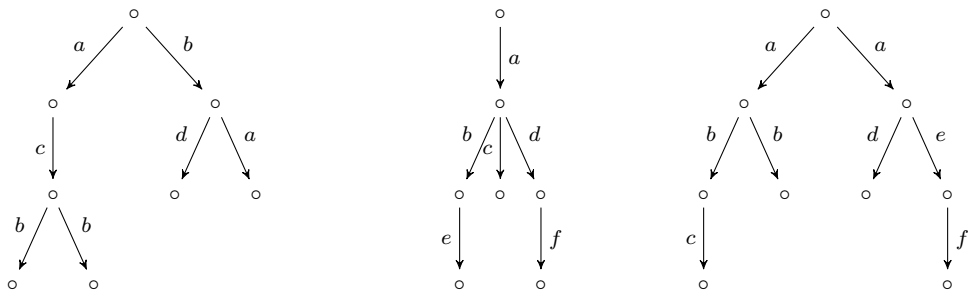


Figure 1.6: Operational description and syntax in *BCCSP*.

terization of bisimulation for finitary processes, the nowadays famous *Hennessy-Milner logic*.

***Definition* 4** *[Hennessy-Milner logic, HML] The set $\mathcal{L}_{HM}$ of Hennessy-Milner logical formulas is defined by: for any finite index set I, if φ, $\varphi_i \in \mathcal{L}_{HM}$ $\forall i \in I$ and $a \in Act$, then we have $\bigwedge_{i\in I}\varphi_i$, $a\varphi$, $\neg\varphi \in \mathcal{L}_{HM}$.*

For each labeled transition system $\mathbb{P}$, the satisfaction relation $\models \subseteq \mathbb{P} \times \mathcal{L}_{HM}$ is defined by:

- *$p \models a\varphi$ if there exists $q \in \mathbb{P} : p \xrightarrow{a} q$ and $q \models \varphi$;*
- *$p \models \bigwedge_{i\in I}\varphi_i$ if for all $i \in I : p \models \varphi_i$.*
- *$p \models \neg\varphi$ if $p \not\models \varphi$.*

Note that $\bigwedge_{i\in\emptyset}\varphi_i \in \mathcal{L}_{HM}$, and so we have $p \models \bigwedge_{i\in\emptyset}\varphi_i$, for all p. Therefore, this is just syntactic sugar for $\top$.

R. J. van Glabbeek included in his encyclopedic work a logical characterization for each semantics in the obtained spectrum, by means of fragments of *HML*. To be exact, he also considered infinitary processes and for that he needed to extent the logic with infinite conjunctions. Moreover, R.J. van Glabbeek also introduced some syntactic sugar in his formulas, which made necessary to extend the definition of the satisfaction relation. Next, we recall the logical characterization for the main semantics mentioned above:

> Bisimulation. The class $\mathcal{L}_B$ of *infinitary Hennessy-Milner formulas* over *Act* is defined by:
>
> - For any set I, if $\varphi_i \in \mathcal{L}_B$ $\forall i \in I$ then $\bigwedge_{i\in I}\varphi_i \in \mathcal{L}_B$.
> - If $\varphi \in \mathcal{L}_B$ and $a \in Act$ then $a\varphi \in \mathcal{L}_B$.
> - If $\varphi \in \mathcal{L}_B$ then $\neg\varphi \in \mathcal{L}_B$.

Trace semantics. The set $\mathcal{L}_T$ of *trace formulas* over Act is defined recursively by:

- $\top \in \mathcal{L}_T$.
- If $\varphi \in \mathcal{L}_T$ and $a \in Act$ then $a\varphi \in \mathcal{L}_T$.

Readiness semantics. The set $\mathcal{L}_R$ of *ready formulas* over Act is defined recursively by:

- $\top \in \mathcal{L}_R$.
- $X \in \mathcal{L}_R$ for $X \subseteq Act$.
- If $\varphi \in \mathcal{L}_R$ and $a \in Act$ then $a\varphi \in \mathcal{L}_R$.

Failures semantics. The set $\mathcal{L}_F$ of *failure formulas* over Act is defined recursively by:

- $\top \in \mathcal{L}_F$.
- $\overline{X} \in \mathcal{L}_F$ for $X \subseteq Act$.
- If $\varphi \in \mathcal{L}_F$ and $a \in Act$ then $a\varphi \in \mathcal{L}_F$.

The satisfaction relation is extended in this way:

- $p \models \top$ for any $p \in \mathbb{P}$.
- $p \models X$ if $Init(p) = X$, where $Init(p)$ is the set of action that p can initially execute.
- $p \models \overline{X}$ if $Init(p) \cap X = \emptyset$.

1.4 Distances

A distance is a numerical description of how far apart two objects are. Roughly speaking, a distance may refer to a physical length or to any estimation based on other criteria. In mathematics, a distance function (or metric) is a generalization of the concept of physical distance. A metric is a function that describes when the elements of some space are "quite similar" or "close to", or "far away from" each other.

***Definition* 5** *Given a set of elements X, a distance (or metric) over X is any function $d : X \times X \to \mathbb{R}$, with $\mathbb{R}$ the set of real numbers, satisfying the following conditions, for $x, y, z \in X$:*

- *It is non negative: $d(x, y) \geq 0$.*
- *Identity of indiscernible: $d(x, y) = 0$ if and only if $x = y$. Distance is zero precisely for equal points.*
- *It is symmetric: $d(x, y) = d(y, x)$. The distance between x and y is the same in either direction.*

– *It satisfies the triangle inequality:* $d(x,z) \leq d(x,y) + d(y,z)$. *The distance between two points is the shortest distance along any path connecting them.*

A metric space is a pair (X,d), where X is a set in which we have defined a distance d. We can find in the literature some terminology referring to several distance concepts, such as *pseudometric*, which is a metric without the identity of indiscernible property; *quasimetric* which is a metric without the symmetry property; and *hemimetric*, which is a metric without both the identity of indiscernible and the symmetry properties. For each of these cases we can define the corresponding *pseudometric (quasimetric, hemimetric) spaces.*

Among the metrics used in (Theoretical) Computer Science, the *Kantorovich metric* has a special importance for us. We refer the reader to either the paper by Anatoly M. Vershik [Ver06], or by Yuxin Deng and Wenjie Du [DD09], in order to learn about this important metric and its applications. As an appetizer, we will say that this metric appears in connection with "the transportation problem". It has been used in many areas such as probabilistic concurrency, image retrieval, data mining, and bioinformatics (although appearing in the literature under different names). Roughly speaking, the Kantorovich metric provides a way of measuring distance (or dissimilarity) between two distributions of mass in a space that is itself endowed with a ground distance. This is done by transforming each of the individual masses that conforms one of the distribution into those conforming the other. The Kantorovich metric can be computed in polynomial time, reducing it to a particular case of the discrete mass transportation problem. There are several well-known polynomial time algorithms to solve this problem as it is shown in [Orl88].

Next, we recall the mathematical definition of the Kantorovich metric given for a separable metric space (S,d). A metric space is separable if it contains a countable, dense subset (i.e., there exists a sequence $\{x_n\}_{n=1}^{\infty}$ of elements in S such that every nonempty open subset of S, contains at least one element of the sequence).

Next we assume that the reader is familiar with the basic notions from probability and measure theory.

***Definition* 6** *Given two Borel probability measures* $\mathbb{P}$ *and* $\mathbb{Q}$ *on* S*, the* Kantorovich distance *between* $\mathbb{P}$ *and* $\mathbb{Q}$ *is defined by*

$$K(\mathbb{P},\mathbb{Q}) = sup\left\{\left|\int f d\mathbb{P} - \int f d\mathbb{Q}\right| : ||f|| \leq 1\right\}.$$

where $||\cdot||$ *is the* Lipschitz semi-norm *defined by* $||f|| = sup_{x\neq y}\frac{|f(x)-f(y)|}{d(x,y)}$*, for a function* $f : \mathcal{S} \to \mathbb{R}$*, with* $\mathbb{R}$ *being the set of all real numbers.*

The fragment of the paper by Y. Deng and W. Du [DD09] presented in Fig. 1.7, shows the formulation of the transportation problem in mathematical terms. Since we have just copied the original text at that paper, we have also preserved the typo there: $\mathbb{Q}(x_j)$ should be read $\mathbb{Q}(y_j)$ at the third line.

> Many problems in computer science only involve finite state spaces, so discrete distributions with finite supports are sometimes more interesting than continuous distributions. For two discrete distributions $\mathbb{P}$ and $\mathbb{Q}$ with finite supports $\{x_1, ..., x_n\}$ and $\{y_1, ..., y_m\}$, respectively, minimizing the total cost of a discretized version of the transportation problem reduces to the following linear programming problem:
>
> $$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{n}\sum_{j=1}^{m}\mu(x_i,y_j)d(x_i,y_j) \\ \text{subject to} & \bullet\ \forall 1 \le i \le n : \sum_{j=1}^{m}\mu(x_i,y_j) = \mathbb{P}(x_i) \\ & \bullet\ \forall 1 \le j \le m : \sum_{i=1}^{n}\mu(x_i,y_j) = \mathbb{Q}(x_j) \\ & \bullet\ \forall 1 \le i \le n, 1 \le j \le m : \mu(x_i,y_j) \ge 0. \end{array} \quad (1)$$

Figure 1.7: The transportation problem in [DD09].

If we have two distributions $\mathbb{P}$ on $\{x_1, \ldots, x_n\}$ and $\mathbb{Q}$ on $\{y_1, \ldots, y_m\}$, the problem of transforming $\mathbb{P}$ into $\mathbb{Q}$ minimizing the total cost, when the cost of transforming a unit in x_i into a unit in y_j is $d(x_i, y_j)$, can be represented as the transportation problem in Fig. 1.7, for the particular case in which we have $\mathbb{P}(x_i) \ge 0$, $\sum \mathbb{P}(x_i) = 1$, $\mathbb{Q}(y_j) \ge 0$, and $\sum \mathbb{Q}(y_j) = 1$. There, the unknowns $\mu(x_i, y_j)$ represent how many units will be transported from x_i to y_j, i.e. in terms of the original distance problem, what will be the optimal solution producing $K(\mathbb{P}, \mathbb{Q}) = \sum_{i=1}^{n}\sum_{j=1}^{m}\mu(x_i, y_j)d(x_i, y_j)$. Therefore, the computation of $K(\mathbb{P}, \mathbb{Q})$ is reduced to the resolution of the linear programming problem presented in (1).

In this thesis, the notion of distance is the topic of Chapter 6 and Chapter 7. This part of the thesis deals with the problem of finding "how far away" two processes are with respect to a given semantics. Obviously, we expect that two equivalent processes are at distance 0.

1.5 This thesis

This thesis emerged from my interest in learning some of the formalisms that are in the basis of Theoretical Computer Science. During the final year of my degree in Mathematics, I started the collaboration with my supervisor D. de Frutos Escrig. He recommended me to start reading the textbook by M. Hennessy [Hen88],

which awakened my curiosity in this area. Soon I started a detailed reading of the "classification of semantics" written by R. J. van Glabbeek [vG01].

The seed of process semantics was planted in my brain, and it received "the water needed" to grow up. I had the opportunity of attending to the presentation of the thesis by C. Gregorio Rodríguez (a former Ph.D. student of my supervisor) [GR09]. There, I received a global vision of his work in this field, learning that (bi)simulations are the cornerstone to capture any of the semantics in the *ltbt-spectrum*. Using them we can provide a coinductive, (bi)simulation-like characterization of these semantics [dFG09]. Also, I acquired the knowledge of the theoretical framework that gives the power to define in an uniform way any (reasonable) semantics preorder between processes [dFG08b]. Once this framework was stablished, and thanks to the crucial role of constrained simulation, D. de Frutos Escrig, C. Gregorio Rodríguez and M. Palomino presented both the observational [dFGP09b] and the equational unified frameworks [dFGP09a].

These papers were the starting point of my scientific work. As soon as I understood the theory that was developed in order to unify, in a very nice way, the semantics in [vG01], and following the suggestions of my supervisor, we started to work in the logical characterization of these semantics. It was also the topic of my master's thesis [Rom10], where we presented a previous study of these characterizations (later revised and improved in our first publication [RdF11]). Although R. J. van Glabbeek collected in his work a logical characterization of all of the semantics in his *spectrum*, it was clear to us that they have been obtained one by one. Thus some no trivial work was to be done in order to find the elements to develop a unified presentation of all of the semantics. Once we had that unified logical characterization, it was time to put everything together: I collaborated with C. Gregorio-Rodríguez, M. Palomino, and of course with my supervisor, to write an extensive paper which collected all our unification work, relating the results in the observational, the equational, and the logical frameworks.

Along this search, we naturally raised the question of *What more can we say when two processes are not equivalent?*, and the concept of distance between processes appeared. Certainly, some other authors such as Thomas A. Henzinger et al. [CHR10, CHR12] and Uli Fahrenberg et al. [FLT11, FTL11], have already followed this path by presenting several distances, mainly based on the idea of quantifying the bisimulation game. But, we considered that these distances did not express accurately the real distance between the processes. Hence, we looked for a new notion that would satisfy our requirements. We found that it could be based on the transformation of trees by adequate weighted rewriting rules, taking into account that trees are the canonical representations of bisimilarity classes.

Now, the path followed since I started my thesis five years ago, in 2011, has

been sketched. Mainly we followed two different research directions that however are clearly related: the study of notions around the formal definitions of equivalences and dissimilarities between processes.

1.5.1 Overview of my objectives

Obviously, this thesis can be included into the general study and classification of process semantics. More specifically, we focus on the study of the relationships between processes. We start from bisimulation, the main notion among these relations, and then we see how to extend the obtained results to the rest of the semantics developed along the last years.

As we said, we were interested in the study of the logical formulas characterizing each semantics in the *ltbt-spectrum*. In [GR09] my colleague from *UCM*, C. Gregorio Rodríguez, together with my supervisor, provided a unified observational and equational theory of all the semantics in the *spectrum*, that will be presented in Section 2.3. The underlying ideas were the isolation of the required observations in each case, and the notion of constrained simulation. These ideas have been essential in my first contribution in order to complete the work of getting the logical characterization of the semantics. In this way our first goal was born, and the following starting point in my trajectory as a researcher was fixed.

Objective 1: Provide a unifying logical framework to the unified spectrum, completing in this way the previous work done in [GR09].

In particular, we overhaul the (unified) axiomatization of the different semantics, that is the basis of our logical unification. For further details about these issues, we refer the reader to the work in [GR09] or, as an extensive summary, to the publication [dFGPR13] in Section 5.2.

The second major pathway of our research came naturally when we concluded the unification work. The scientific community already had a full unified view of the *spectrum* in three frameworks: observational, axiomatic and logical. Thus, the next step was obvious: *How can we measure dissimilarity between processes?*. Consequently, the Section 2.4 is devoted to the presentation of the basic notions around the definition of distances between processes.

We went along the extensive literature on this topic, including the recent thesis by Arjun Radhakrishnan, from the Institute of Science and Technology Austria, in Klosterneuburg (Austria) [Rad14], where our first papers on distances have been already cited. We also reviewed several "classical" notions of distances that always use the weighted bisimulation game as basis to define them. We surfed through the work by Uli Fahrenberg, Axel Legay, and Thomas A. Henzinger,

among others, and these studies led us to our second goal, which can be divided into two parts:

Objective 2.1: Developing a new notion of distance which, in our opinion, would fulfill the original requests in order to adequately measure dissimilarity.

Objective 2.2: Extending this concept to the case of infinite (but finitary) processes.

For reasons that will be explained in Chapter 2, in order to capture the differences between the behaviors of two systems, we decided to introduce a new notion of distance. We think that this notion gives a "more realistic" measure than the others presented in the previous work in the literature.

1.5.2 Thesis structure

This thesis has been developed under the format of a collection of publications.

Among the chapters of this thesis, apart from the introduction, we will present first our preliminaries –Chapter 2– that contents the seminal ideas and basic concepts supporting this thesis, giving us a better understanding of the technical details in it. We will start with the main concepts in the world of process semantics. Afterwards, we have two main parts reflected in its two sections. In the first one, Section 2.3, we will review the preliminary work on the unification of semantics for processes (developed by my predecessor under the guidance of my supervisor). The second, focusses on a (reasonably) deep study of distances between processes based on the bisimulation game. Furthermore, in Section 2.4 we also include some work on distances between probabilistic systems by Kim Guldstrand Larsen et al. and Franck van Breugel et al. Although we still have no published results on distances for probabilistic systems, recently we started to work in this topic, and we hope to have soon a probabilistic version of our approach to define distances between processes. This is one of the reasons for including those papers in the package of related work.

Chapter 3 presents our results and the existing relationship between the publications that make up this thesis. To end the first part of it, in Chapter 4 we briefly summarize our scientific contributions to the field of Process Semantics and present some (close and far) future work. Next Part II, it contains the three chapters with our publications. Chapter 5 collects our papers completing the unification work between process semantics, providing the logical framework; Chapter 6 starts our study in the subject of distances between processes, by presenting the finite case; Chapter 7 presents our coinductive definition, that makes

possible to compute the distance also between infinite processes. Furthermore, we present our result proving the equivalence between the finite and the coinductive definition of finite processes. Part III of our thesis, collects some missing proofs in our papers, that due to space reasons could not be published in the sent version to the conference. Additionally, we present there some small bugs that could obscure the comprehension of the paper. Please if you think that something is wrong when reading our papers, check this in our Chapter 8. Perhaps it could be a bug that has been already detected, if this is not the case, we apologize for our non-detected mistakes when reviewing the papers.

The chapters collecting publications will begin with a short comment about the forum where they were presented. Next, divided into sections within the chapter, the full text is included, respecting the formats, headers and styles of the original publications.

This work includes at the beginning of each chapter different fragments of the most important literary Spanish work: *El Ingenioso Hidalgo Don Quijote de La Mancha –The Ingenious Gentleman Don Quixote of La Mancha–*, written by Miguel de Cervantes, as a wink to the country and citizenship of this humble author.

Chapter 2

Preliminaries

"Somewhere in La Mancha, in a place whose name I do not care to remember..."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter-1 Part-1

"En un lugar de la Mancha, de cuyo nombre no quiero acordarme..."

This chapter introduces the fundamental concepts which have made possible the research collected in this thesis. As we have briefly seen Computer Science has undergone a quick evolution from its origins. Bit by bit, it has become an important and essential tool in everyday life. Nowadays, it is practically impossible to find someone who does not have a mobile phone, a tablet, or a laptop. Thus, the so called computer errors (or bugs) such as network outages, unexpected behaviors, or failures in communication systems, maintain the world in a totally undesired "standby" state. Therefore, Theoretical Computer Science is a very important area of study in order to help us in the understanding of how computers work and how they could evolve. As far as we make progress in this field, we will get more and more powerful, and specially more secure systems.

This thesis can be classified in the field of Theoretical Computer Science, more specifically within the study of process semantics. To be more precise, our research focuses on the relationships between these processes. This is why we decided to start this chapter with a section devoted to the basic aspects of the theory of process semantics and their presentations. After that, we will present in Sections 2.3 and 2.4 some previous work in the unification of the process semantics and distances between processes. After this introductory chapter, we will have the necessary tools to present the results obtained in this thesis –Section 3–.

2.1 Process semantics

As we have seen in the previous chapter, process semantics theory emerged in the late sixties of the last century. Edsger W. Dijkstra was a pioneer in the study of process semantics [Dij68a] (reedited in [Dij02]) mainly studying in the beginning sequential processes [Dij65, Dij68b]. Some years later, arrived the formalism sketched in Section 1.2, and the first complete model of process: *CCS*, developed by Milner in his book *A Calculus of Communicating Systems* [Mil80].

Over the years, process semantics evolved, and new tools were added to the models. Thanks to them we faced more complicated parameters, such as time, probabilities, and priorities. As we had already briefly dealt with these topics in the previous chapter, and it is not our intention to provide here all the details about process semantics, next we will mainly concentrate on the modeling of operational semantics.

2.1.1 Labeled transition systems and their variants

We have seen in Section 1.3 that *labeled transition systems (lts)* have been used to model the operational semantics of processes. Labeled transition systems are an elegant way of separating the operations (actions) from the computational states

of a system. As far as we know, one of the founding fathers of lts was Robert M. Keller. In his paper [Kel76] he used the name *Named Transition Systems* for referring to lts –See Fig. 2.1–.

1. The Conceptual Model

The conceptual model we present appears to be general enough to capture most notions of parallel computation which have been proposed. Its usefulness is not so much as a practical tool, but, as the name indicates, as a way to conceive of the operation of a system for parallel computation and of the properties of such a system. Formally, we call this model a "transition system."

Definition 1.1. A *transition system* is a pair $(Q, \rightarrow)$, where Q is a set of *states* and $\rightarrow$ is a binary relation on Q, called the set of *transitions*.

We do not restrict either Q or $\rightarrow$ to be finite. When q and q' are states, we think of $q \rightarrow q'$ as meaning that there can be an *indivisible* progression from state q to state q'. In a system which allows "parallelism," for a given state q there may well be many states q' such that $q \rightarrow q'$. That is, $\rightarrow$ is not necessarily a function. Each such state represents a result of one of a number of possible *actions*, each of which is *simultaneously enabled* when the system is in state q.

It is appropriate to extend our definition of transition system as follows.

Definition 1.2. A *named transition system* is a triple $(Q, \rightarrow, \Sigma)$ where $(Q, \rightarrow)$ is a transition system and each transition is assigned one or more names in the set Σ.

According to the preceding definition, a named transition system can be *presented*, in the case where Σ is finite (or, more generally, recursive), by presenting a binary relation on Q, $\rightarrow^t$, for each t in Σ. In other words, $q \rightarrow^t q'$ means that there is a transition from q to q' with name t, and $\rightarrow$ is just $\bigcup \{\rightarrow^t \mid t \in \Sigma\}$. By the arbitration condition, we may assume that the transitions so named represent atomic, indivisible actions.[1]

Figure 2.1: The definition of *lts* as it was presented in [Kel76].

This model was deeply studied and developed by G. D. Plotkin in his technical report titled *A structural approach to operational semantics* [Plo81, Plo04], as he explains in his Introduction:

> It is the purpose of these notes to develop a simple and direct method for specifying the semantics of programming languages.

The process algebra *BCCSP* syntactically describes finite lts, that is, those where the set of states Q is finite, and we cannot return to the same state via a transition sequence. The definition of *BCCSP* syntax and its operational rules were respectively presented in Def. 2 and Def. 3 –see page 16–. *BCCSP* has been extensively used in the literature in process theory –see e.g. [vG01, AFI07, AdFGI14]–. Specially its use should be pointed out in these last works, where the focus is on the study of semantics properties.

When new properties or characteristics were added to the models, new ways of modeling the operational semantics of a process appeared. Next, we briefly present some of them.

Kripke Structure (KS). It was probably the first variant of transition system. As a matter of fact, more than a variant it was a precursor of lts, since it was

proposed by Saul Kripke in the beginning of sixties [Kri63]. He added, to each state, the set of atomic properties satisfied in it. Given a set of atomic properties AP, we can define a KS as a tuple $\mathcal{A} = (X, \rightarrow, L)$ where $(X, \rightarrow)$ is a transition system and $L : X \rightarrow \mathcal{P}(AP)$ is the so called observation (or labelling) function, that maps each state to the set of atomic properties satisfied in it.

Note that Kripke structures have no labels at the transitions (although they could be "inserted" at the labels of the states in a tricky way). The work on this topic started long time ago with the contribution by Rocco de Nicola and Frits W. Vaandrager [DV90]. There are also some recent contributions, as that by Michel A. Reniers and Tim A. C. Willemse [RW11]. However, when defining *lts* the focus was not on the states, but on the transitions. Hence, the main intention of lts is to capture the dynamics of the systems which is represented by their transitions.

Alternating Transition System (ATS). These structures are multi-agent systems [HF89], in which the transitions are fixed by these agents. ATSs were introduced by Rajeev Alur et al. in [AHK97] and they modeled the *input-output* behaviors of the systems. Formally an ATS is defined by a tuple

$$\langle Proc, (Proc_{in}, Proc_{out}), Act, (Act_{in}, Act_{out}), \rightarrow \rangle,$$

where $(Proc, Act, \rightarrow)$ is a *lts*; $Proc_{in}$ and $Proc_{out}$ are called respectively the input and output sets of states, and form a partition of $Proc$ (that is, $Proc_{in} \cup Proc_{out} = Proc$ and $Proc_{in} \cap Proc_{out} = \emptyset$). Act_{in} and Act_{out} (respectively the input and output alphabets) form a partition of Act; and the transitions are represented by $\rightarrow \subseteq \{Proc_{in} \times Act_{in} \times Proc\} \cup \{Proc_{out} \times Act_{out} \times Proc\}$. Therefore, the transitions from input states correspond to input actions, while the transitions from output states correspond to output actions.

Probabilistic Labeled Transition System (PLTS). They were defined by Kim G. Larsen et al. at the end of the eighties [LS89a] (later reedited in [LS91]). As their name indicates, PLTSs introduce probabilistic information in *lts*. They are formally defined by a tuple $(Proc, Act, t)$, where $Proc$ is the set of processes, Act is the set of actions and $t : Proc \times Act \times Proc \rightarrow [0, 1]$ is the probabilistic transition function satisfying that for all $p \in Proc$ and $a \in Act$ $\sum_{p' \in Proc} t_{p,a}(p') \leq 1$.

In the literature these systems are also called partial labeled Markov chains (see, for instance [DGJP99]).

Weighted Labeled Transition System (WLTS). They were initially studied by Bartek Klin in [Kli09] (extended in [KS13]). Formally a WLTS is defined by a tuple $(Proc, Act, \rho)$, where $Proc$ is the set of processes, Act is the set of actions and $\rho : Proc \times Act \times Proc \rightarrow W$ is the so called *weight function*. We take as W an adequate set, which typically depends on the characteristics of the

transitions that we want to capture. To make the notation similar to that for ordinary *lts*, we simply write $p \overset{a,w}{\to} p'$, when $\rho(p \overset{a}{\to} p') = w$.

The weights in a transition system are used to represent some quantitative metric associated to transitions. Recently, these extensions have been combined, getting the *Weighted Probabilistic Transition Systems (WPTS)* [CCH$^+$11], which consist of a PLTS and a weight function $v : \mathit{Proc} \times \mathit{Act} \times \mathit{Proc} \to \mathbb{Q} \cup \{\infty\}$ that assigns weights to probabilistic transitions.

Of course, there are many other ways of modeling the operational semantics. This is the case for *Petri nets* [Pet73], which model concurrency by means of events cooperating on a common task; or *Automata theory* [Fau82, LS89b], which has also been used to study the semantics of systems. As a matter of fact, at the syntactical level an automaton is just a *lts*, where a set of (final) states is distinguished to point out (in some way) correct termination. Nevertheless, automata theory is (semantically) focused on the study of generated (or accepted) languages, thus it only covers the trace semantics of processes. Even so, there are some contributions that combine language theory and process algebra, as that by Jos C. M. Baeten et al. [BCLvT09].

But since we will not use at all these interesting frameworks in this thesis, we will not say anymore about them here.

2.2 Relationships between processes

Informally, a process is a model of the behavior of a system. Systems can perform actions (which correspond to the transmission of some objects, or just to the transformation of the system itself). The models of processes define their behaviors. Certainly, we are interested in how processes evolve along their executions. But, in order to study the mathematical properties of processes, and therefore develop a good theory for them, we need a more sophisticated notion, further than the sole enumeration of their behaviors.

By defining an equivalence relation over a set, we establish a relationship between the elements in this set: it is expected that those equivalent elements share some basic properties (or characteristics). In particular, when talking about processes we are interested in capturing in a suitable way the notion of having the same (or equivalent) behaviors. Consequently, by means of an equivalence relation, we will get a notion which indicates when two elements of the corresponding set can be viewed as different "faces" of the same "individual". In the case of processes, we can exchange two of them whenever they have the same meaning, without altering the behavior of the system containing such processes as a component. Sometimes we will go further, and we will define an order relationship

instead of an equivalence. In this case, the aim is to formalize some "natural" order between the elements in a set, by using a binary relation. Every order relation induces an equivalence (its so-called kernel), so that the former generalizes the latter. In our case, the importance of order relationships is that they can reflect essential properties such as *to be an implementation of*, or *to be faster than*, or *to be safer than ...*

Within the framework of process semantics the equivalence between processes was an important topic since the beginning. The work by R. Milner [Mil70a] presents several equivalence relations between program schemata. Later, C. A. R. Hoare [Hoa78] compares the equivalence relationships for *CSP* with those for *CCS*. Quoting Hoare:

> In general, equality in CCS is a strong relation, since equal processes must resemble each other both in their observable behavior and in the structure of their hidden behavior. CCS is therefore a good model for formulating and exploring various weaker definitions of equivalence, which ignore more aspects of the hidden behavior. Milner accomplishes this by introducing the concept of observational equivalence.

The notion of equality in *CCS* was formalized by Milner in [Mil80] by the following definition of observational equivalence.

***Definition* 7** *Let be $\mathcal{A} = (W, A, \rightarrow)$ a LTS. Given a set W of states, we define the class of observational equivalence relations $\sim_n \subseteq W \times W$, for each $n \in \mathbb{N} \cup \{\omega\}$ as follows:*

- $\sim_0 = W \times W$.
- $s \sim_{n+1} t$ *for* $n \geq 0$ *if:*
  1. *for all s' with $s \xrightarrow{a} s'$, there exists t' such that $t \xrightarrow{a} t'$ and $s' \sim_n t'$, and*
  2. *for all t' with $t \xrightarrow{a} t'$, there exists s' such that $s \xrightarrow{a} s'$ and $s' \sim_n t'$.*
- $\sim_\omega = \bigcap_{n \geq 0} \sim_n$.

***Remark* 1** *The intuitive idea Milner tried to capture is that whenever we have $p \sim_n q$, then they could not be distinguished by any sequence of at most n actions, and thus $p \sim_\omega q$ it would be impossible to distinguish them at all.*

However, it was not until D. Park presented the notion of bisimulation [Par81] –see Fig. 2.2–, which we will show below, that a totally satisfactory way of capturing this intuition appeared.

Firstly, we give definitions for the notion applied to finite automata $M = \langle S, s_o, M, F\rangle$, $M' = \langle S', s_o', M', F'\rangle$.

Say that M simulates M' via R - in symbols $M \underset{R}{\leadsto} M'$, or just $M \leadsto M'$ - if $R \subseteq SxS'$, and, writing $s \leadsto s'$ for $\langle s,s'\rangle \in R$,

1) $s_o \leadsto s_o'$

2) $s \in F,\ s \leadsto s' \Rightarrow s' \in F$

3) $\sigma \in \Sigma \cup \{\lambda\},\ s_1 \leadsto s_1',\ s_2 \in M(s_1,\sigma)$
$\Rightarrow s_2 \leadsto s_2'$ for some $s_2' \in M'(s_1',\sigma)$

Say that M bisimulates M' via R (in symbols $M \underset{R}{\leftrightsquigarrow} M'$, etc.) if M simulates M' via R, and M' simulates M via $R = \{\langle s',s\rangle \mid \langle s,s'\rangle \in R\}$.

Figure 2.2: Bisimulation notion as it appeared in [Par81].

2.2.1 Bisimulations and simulations

As said above, the notion of bisimulation was initially presented by D. Park [Par81]. Next, we recall the definition of bisimulation between the states of a *lts*.

***Definition* 8** *A binary relation R over the states of a* lts *is a bisimulation, if $(p,q) \in R$ implies that for every $a \in Act$,*

- *If $p \xrightarrow{a} p'$ then, there exist q' such that $q \xrightarrow{a} q'$ and $(p',q') \in R$.*
- *If $q \xrightarrow{a} q'$ then, there exist p' such that $p \xrightarrow{a} p'$ and $(p',q') \in R$.*

The union of all the bisimulations is called bisimilarity*, and we will write $p \simeq q$ whenever p and q are related by some bisimulation.*

Intuitively, two systems are bisimilar if they match each other's moves. In this sense, each of the systems cannot be distinguished from the other by an external observer. In order to prove $p \simeq q$ it is enough to present a relation R that contains the pair (p,q), and prove that R is really a bisimulation in the sense of Def. 8 (by checking that each pair satisfies the conditions in this definition).

Certainly, the only difference between Def. 7 and Def. 8 appears when we consider infinitely branching systems. In this case, the coinductive approach becomes crucial in order to capture the desired equivalence relation. Instead, the inductive approach in Def. 7 fails to do that, because in general inductive techniques

cannot deal with the singular situations that appear when König's lemma fails at infinitary trees. The advantages of Def. 8 were soon recognized even by Milner (and the whole scientific community) incorporating it to his theory [Mil89].

In many cases, an equivalence relation can be characterized by means of a set of axioms. Next, we recall the so called bisimulation axioms, that characterize the bisimilarity relation (for BCCSP):

$$(B1)\ x + y \simeq y + x \qquad (B2)\ x + x \simeq x$$
$$(B3)\ (x + y) + z \simeq x + (y + z) \qquad (B4)\ z + \mathbf{0} \simeq z$$

But the real utility of bisimulations arrives when we consider infinite systems. In this case, coinduction appears as a tricky way of considering the property to be proved when making a circular (but totally satisfactory!) reasoning. The power of this approach is immediately recognized, sometimes in connection with category theory, that supports it in a very general way.

As a consequence, over the last years there has been a great amount of work developing the semantics in a coalgebraic framework [BG03, HJ04, Kli04, Jac04, FPdF07]. Bisimulation can be seen as the adequate way of describing the equality of objects defined by a coalgebra. A good review about the history and applications of bisimulation and coinduction can be found in the books *Introduction to Bisimulation and Coinduction* by Davide Sangiorgi [San11], and also the paper *Advanced Topics in Bisimulation and Coinduction* by D. Sangiorgi and J. Rutten [SR11].

Among the alternative characterizations of bisimilarity we distinguish here the Hennessy-Milner Logic [HM85] –See Def. 4 in page 18–. We have that two processes are bisimilar if and only if both satisfy the same formulas in *HML*. Another very useful characterization is provided by seeing bisimulation as a game. The bisimulation game was formally defined in [Tho93] and later particularized in [NC94, Sti98].

***Definition* 9 (Bisimulation Game)** *Given $p, q \in Proc$, we call configurations the pairs (p', q'), with $p', q' \in Proc$. The bisimulation game is played by two players: the attacker $\mathbb{A}$ and the defender $\mathbb{D}$. The initial configuration of the game deciding if $p \simeq q$, is just the pair (p, q). A round of the game, when the current configuration is (p', q'), proceeds as follows:*

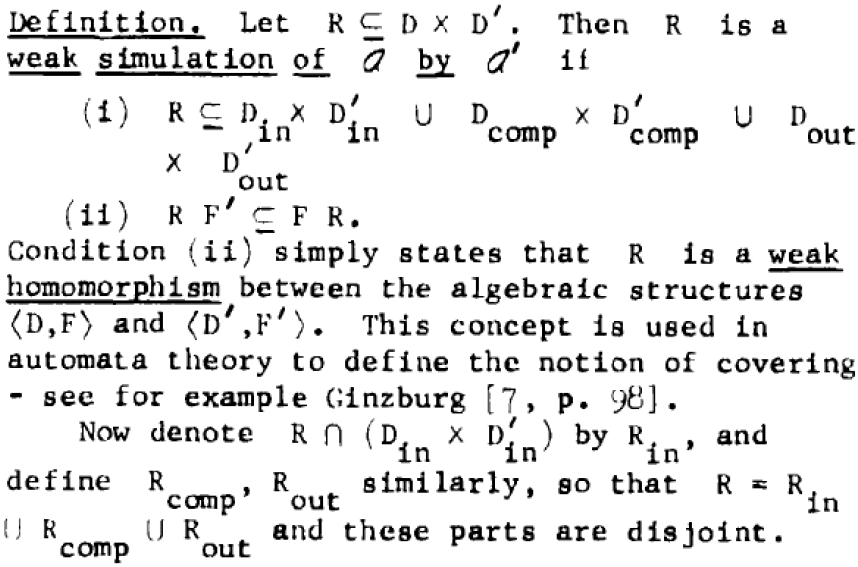
1. *$\mathbb{A}$ chooses a transition, either in p' or q'. Let us suppose it is p': $p' \xrightarrow{a} p''$.*
2. *$\mathbb{D}$ must execute the same action at the other side of the board (in this case q'): thus choosing q'' with $q' \xrightarrow{a} q''$.*

3. The game proceeds in the same way from the new configuration (p'', q'').

The winner of the game is defined by the following rules: (1) Any infinite game is a win for $\mathbb{D}$. *(2)* $\mathbb{D}$ *also wins if* $\mathbb{A}$ *cannot make any new move. (3)* $\mathbb{A}$ *wins when it makes a move that* $\mathbb{D}$ *cannot reply.*

When playing the bisimulation game, we have that p and q are bisimilar if and only if there exists a winner strategy for the defender $\mathbb{D}$. If instead we have a winner strategy for the attacker $\mathbb{A}$, we have proved that p and q are not bisimilar.

Milner also defined the concept of simulation [Mil70b, Mil71]. Roughly speaking, we can say that a simulation is a half of a bisimulation: a process p simulates q, if p is able to do the same things that q can do. Thus, in the framework of simulations we have a fixed process (q) that executes actions, while the other (p) tries to mimic them. Mathematically, as Milner said, the concept of simulation is related with the algebraic concept of homomorphism.


Definition. Let $R \subseteq D \times D'$. Then R is a weak simulation of a by a' if

(i) $R \subseteq D_{in} \times D'_{in} \cup D_{comp} \times D'_{comp} \cup D_{out} \times D'_{out}$

(ii) $R\,F' \subseteq F\,R$.

Condition (ii) simply states that R is a weak homomorphism between the algebraic structures $\langle D,F\rangle$ and $\langle D',F'\rangle$. This concept is used in automata theory to define the notion of covering - see for example Ginzburg [7, p. 98].

Now denote $R \cap (D_{in} \times D'_{in})$ by R_{in}, and define R_{comp}, R_{out} similarly, so that $R = R_{in} \cup R_{comp} \cup R_{out}$ and these parts are disjoint.

Figure 2.3: The definition of simulation as it was presented in [Mil71].

***Definition* 10** *A binary relation* S *over the states of a* lts *is a simulation, if* $(p, q) \in S$ *implies that for every* $a \in Act$,

– *If* $p \xrightarrow{a} p'$ *then, there exist* q' *such that* $q \xrightarrow{a} q'$ *and* $(p', q') \in S$.

The union of all the simulations is called the similarity *relation, and we write* $q \sqsubseteq p$ *(or* $p \sqsupseteq q$*) whenever* p *simulates* q *(i.e. there exists a simulation* S*, such that* $(p, q) \in S$*).*

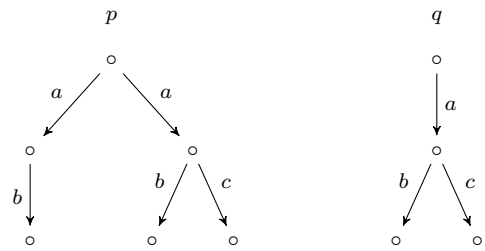


Figure 2.4: Mutual simulation differs from bisimulation.

However, despite the similarities between the definitions of bisimulation and simulation, the differences between the two notions are also quite important: similarity is farther away from bisimilarity than apparently seems. Firstly, similarity is an order relation and not an equivalence (in general). Furthermore, while our informal justification of its definition might let us think that the kernel of similarity should be bisimilarity, this is not at all the case. Of course, the simulation kernel is an equivalence, but it distinguishes much less pairs of processes than bisimulation does. It is remarkable the fact that there is no (non trivial) preorder whose kernel will be the bisimulation equivalence.

It is known that, a mutual simulation is not always a bisimulation. As a well known example, consider the processes in Fig. 2.4. Clearly $p \sqsubseteq q$ and $q \sqsubseteq p$, but if we play the bisimulation game using these two processes, we have that the attacker can first play in p by firing the a-action in the left, and then the defender must reply a in q. Afterwards, the attacker moves in q doing c, and the defender cannot reply in p, so that he loses and we have $p \not\simeq q$.

As done for the case of bisimulation, we can also axiomatize the similarity preorder over $BCCSP$ processes. It coincides with the order induced by the axioms $\{B_1, B_2, B_3, B_4, S\}$, where $\{B_1, B_2, B_3, B_4\}$ are the bisimulation axioms and we have the simulation axiom

$$(S) \quad p \sqsubseteq p + q.$$

There is also a simulation game defined as follows:

***Definition* 11 (Simulation game)** *Given $p, q \in Proc$, we call configurations the pairs (p', q'), with $p', q' \in Proc$. The simulation game is played by two players: the attacker $\mathbb{A}$ and the defender $\mathbb{D}$. The initial configuration of the game deciding if q simulates p (denoted by $p \sqsubseteq q$), is just the pair (p, q). A round of the game, when the current configuration is (p', q'), proceeds as follows:*

1. *$\mathbb{A}$ chooses a transition in p: $p' \xrightarrow{a} p''$.*

2. $\mathbb{D}$ *must execute the same action at the other side of the board (q):* $q' \xrightarrow{a} q''$.

3. *The game proceeds in the same way from the new configuration* (p'', q'').

The winner of the game is defined by the following rules: (1) Any infinite game is a win for $\mathbb{D}$. *(2)* $\mathbb{D}$ *also wins if* $\mathbb{A}$ *cannot make any new move. (3)* $\mathbb{A}$ *wins when it makes a move that* $\mathbb{D}$ *cannot reply with a transition from* q.

Analogous to the bisimulation game, when playing the simulation game, we have that q simulates p if there exists a winner strategy for the defender $\mathbb{D}$. If instead, we have a winner strategy for the attacker $\mathbb{A}$, then we have proved that q does not simulate p.

2.2.2 Trace, Failures, Readiness and other semantics

We will start by focusing on the trace semantics, which is certainly the coarsest non-trivial semantics, and whose importance lies in its simplicity. It collects the sequence of actions that a process is able to execute along any of its possible computations. Next, we will review some other semantics finer than trace (i.e, differencing more processes): Failures (F), Readiness (R), Ready trace (RT), Failure trace (FT) and Ready simulation (RS).

Trace

We begin by quoting an informal definition of *Trace* given by C. A. R. Hoare in [Hoa85]

> A trace of the behavior of a process is a finite sequence of symbols recording the events in which the process has engaged up to some moment in time. Imagine there is an observer with a notebook who watches the process and writes down the name of each event as it occurs. We can validly ignore the possibility that two events occur simultaneously; for if they did, the observer would still have to record one of them first and then the other, and the order in which he records them would not matter.

Formally, we can define the *Traces* of a process as follows:

***Definition* 12** *We define the* traces *of a process over BCCSP by structural induction, as follows:*

- $\mathbf{traces}(\mathbf{0}) = \{\langle\rangle\}$.

- $\mathbf{traces}(ap) = \{\langle\rangle\} \cup \{\langle a\rangle \frown t \mid t \in \mathbf{traces}(p)\}$.
- $\mathbf{traces}(p + q) = \mathbf{traces}(p) \cup \mathbf{traces}(q)$.

where $\langle\rangle$ is the empty trace, and the operator $\frown$ represents the concatenation of symbols chains, as it appears in [Hoa85].

Trace semantics is the simplest semantics, thus over the years there have been a large amount of work devoted to it, such as [Dil89, Jon89, Jon94, NV09]. Under this semantics we cannot distinguish processes such as $a+ab$ and ab, because both define the same set of traces: $\{\langle\rangle, \langle a\rangle, \langle a, b\rangle\}$. However, we could think that these two processes should be distinguished. In particular, the former can become stuck after executing an action a, while the latter always does action b after an a. Thus, we need to improve the observations done along the execution of processes. By observing more details, we get new distinction capacities. In particular, next we will briefly present the following (linear) semantics, where more and more pairs of processes are distinguished.

Failures

A first approximation in order to improve the observations in a process is to consider, apart from the traces, the actions that it cannot execute. Thus, we define a set of pairs (σ, F), where σ is a trace and F is a set of actions. We say that the pair (σ, F) is a *failure* of process p if p can execute σ reaching a state where it cannot do any action in the set F. C. A. R. Hoare defined these failures in [Hoa85] under the name of refusal sets, after the famous paper titled *A theory of communicating sequential processes (TCSP)* [BHR84], where the fundamentals on the semantics of *CSP* were stablished.

> The refusal set seems to be the weakest kind of observation that efficiently represents the possibility of nondeterministic deadlock; and it therefore leads to a much weaker equivalence, and to a more powerful set of algebraic laws than CCS.

Again, we can define the failure semantics by using structural induction as follows.

***Definition* 13** *We define the* failures *of a process over BCCSP, by structural induction, as follows*

- $\mathbf{failures}(\mathbf{0}) = \{(\langle\rangle, F) \mid \forall F \subseteq Act\}$.
- $\mathbf{failures}(ap) = \{(\langle\rangle, F) \mid \forall F \subseteq Act - \{a\}\} \cup \{(\langle a\rangle \frown t, F) \mid (t, F) \in \mathbf{failures}(p)\}$.

- $\mathbf{failures}(p+q) = \{(\langle\rangle, F) \mid (\langle\rangle, F) \in \mathbf{failures}(p) \cap \mathbf{failures}(q)\} \cup \{(t, F) \mid t \neq \langle\rangle, (t, F) \in \mathbf{failures}(p) \cup \mathbf{failures}(q)\}$.

Readiness

It is also possible to observe the traces and the actions that a process can run after their execution. Thus, we define pairs (σ, X), where again σ is a trace and X is a set of actions. We say that the pair (σ, X) is a *ready* of process p, if p can execute σ reaching a state where exactly the actions in X can be fired.

Readiness semantics [OH86] is slightly different to failures semantics, and can distinguish more processes. Formally, we can define the readiness semantics as follows:

***Definition* 14** *We define the* readies *of a process over BCCSP by structural induction, as follows*

- $\mathbf{readies}(\mathbf{0}) = \{(\langle\rangle, X) \mid X = \emptyset\}$.
- $\mathbf{readies}(ap) = \{(\langle\rangle, X) \mid \forall X = \{a\}\} \cup \{(\langle a\rangle \frown t, X) \mid (t, X) \in \mathbf{readies}(p)\}$.
- $\mathbf{readies}(p+q) = \{(\langle\rangle, X) \mid X \in Init(p) \cup Init(q)\} \cup \{(t, X) \mid t \neq \langle\rangle, (t, X) \in \mathbf{readies}(p) \cup \mathbf{readies}(q)\}$.

where $Init(p)$ is the set of all actions that p can fire, that is, the set of initial actions in p.

Michele Boreale et al. compared readiness semantics with divergence in testing in [BDP01], studying their differences and similarities.

Ready trace and Failure trace

There are two other semantics which are finer than those presented before: the *Ready Trace semantics* [BBK87] and the *Failure Trace semantics* (for image finite processes, this is exactly the equivalence induced by Iain Phillips notion of *refusal testing* [Phi87]). With these names, it is not so difficult to guess what their underlying idea is: roughly speaking, we can say that they observe the set of readies (or failures) after executing each action, and not just at the end of a trace. Thus, they give us more observational power, and hence we can distinguish between more processes.

DEFINITION 4.4. Let $\mathscr{P} = (\mathrm{Pr}, \mathrm{Act}, \mathrm{Can}, \mu)$ be a probabilistic transition system. Then a $\frac{2}{3}$-bisimulation $\mathscr{R}$ is a binary relation on Pr such that whenever $p\mathscr{R}q$ and $a \in \mathrm{Act}$ then the following hold:

1. Whenever $p \xrightarrow{a} p'$, then $q \xrightarrow{a} q'$ for some q' with $p'\mathscr{R}q'$
2. Whenever $q \xrightarrow{a}$, then $p \xrightarrow{a}$.

Two processes p and q are said to be $\frac{2}{3}$-bisimilar just in case there are $\frac{2}{3}$-bisimulations $\mathscr{R}_1, \mathscr{R}_2$ such that $p\mathscr{R}_1 q$ and $q\mathscr{R}_2 p$. We write $p \simeq q$ in this case.

Figure 2.5: The definition of $\frac{2}{3}$-bisimulation as it was presented in [LS91].

Ready simulation

An interesting and well known order relation close to simulation is *Ready simulation*. Ready simulation is the best known of the constrained simulation semantics, that are simulations whose pairs have to satisfy some additional fixed constraints –see [dFG08a] to learn more about constrained simulations–. As in the case of bisimulation and simulation, ready simulation has appeared in many contexts. On the one hand, it appeared in the original work by Bard Bloom et al. [BIM88]; on the other hand, it was defined in the work by K. G. Larsen [LS91], although there it was called $\frac{2}{3}$-bisimulation –See Fig. 2.5–.

Roughly speaking, the definition of Ready simulation adds to simulation the duty of checking at every step that the two compared processes have the same initial actions. Formally, we can define it as follows:

***Definition* 15** *A binary relation $RS \subseteq BCCSP \times BCCSP$ over processes is a ready simulation if $(p, q) \in RS$ implies that for every $a \in Act$,*

- *If $p \xrightarrow{a} p'$ then, exists q' such that $q \xrightarrow{a} q'$ and $(p', q') \in RS$.*
- *Furthermore, if $q \xrightarrow{a}$ then $p \xrightarrow{a}$.*

All the semantics presented here, and in general those studied over the last years, can be finitely axiomatized, as it was previously done for (bi)simulation semantics. Furthermore, the relationships between the equivalences classes induced by the semantics presented here define the diamond shown in Fig. 2.6. We will present in Section 2.3 more details about the axiomatizations of the different semantics and the relationships between them.

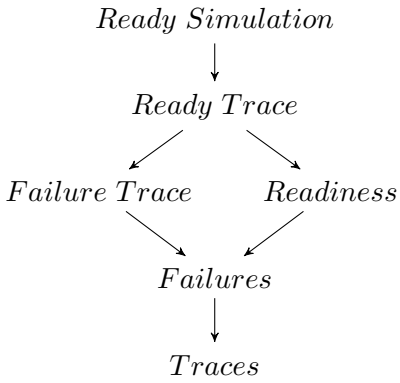


Figure 2.6: The inclusion order between the most popular semantics.

2.2.3 Testing

We use tests for deciding whether a particular process (implementation) is related with another (specification). This notion is a very practical one, and traditionally there has been a strong distinction between testing and proving correctness, in the sense that it is commonly said (Dijkstra, 1972) that

> *Testing can be used to show the presence of bugs, but never to show their absence.*

The starting point is the observation of processes. Hennessy deeply discussed what does it mean to observe a process: a single observation is not enough, since processes are not deterministic. A concrete observation is called a *test*, and it is also possible to consider multiple observers, each of them with a specific behavior. Non determinism gives us a different way of understanding when a test is passed by a process. Thus, we have three possible notions, those known as *must*, *may* and *may-must* testing. These interpretations coincide to the three possible power domains [Plo76]: *must* corresponds with Smyh's power domain, *may* with the power domain by Hoare, and the last one with the domain by Egli-Milner, also called the Plotkin's domain.

Therefore, we can say that two processes are equivalent under testing, if both processes agree with the result of each test, that is, no experiment can find any difference between them. It is also possible to define, in a natural way, the corresponding order relation: *a process p is better than a process q if p passes at least the same proofs as q*. The testing model puts more emphasis on the methodology than in the specific language used to define it:

The general idea of testing processes can be formalized in many different ways and can lead to a variety of interesting equivalences [Hen88].

2.3 Comparing semantics

This section is devoted to shortly present the unification theory developed by my supervisor and one of his previous Ph.D. students, now Assistant Professor, C. Gregorio-Rodríguez [GR09]. They looked for the necessary parameters to properly stratify the spectrum. They decided to read with new eyes the paper by R. J. van Glabbeek [vG01] searching for the pieces that, arranged in a different way, could provided a more precise organization of the *ltbt-spectrum* –see Fig2.7–. In this way, they achieved a better knowledge of the semantics collected there, and of the relationships between them.

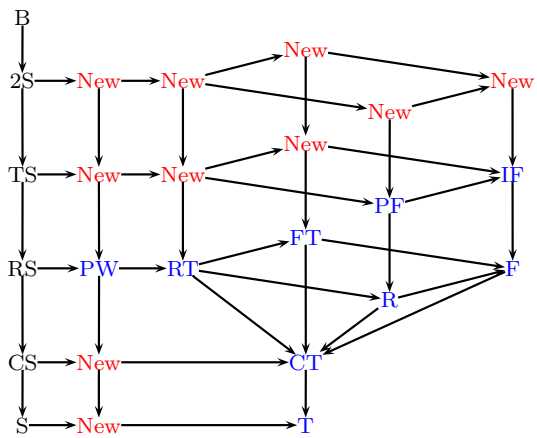


Figure 2.7: The new spectrum presented in [dFGP09a].

This theory has been the foundation upon which we have developed our logical framework. Moreover, the section ends with a quick review of the logical characterizations of the semantics presented by R. J. van Glabbeek in [vG01].

2.3.1 The new unification of the ltbt-spectrum given in [GR09]

Starting from bisimulation, which is an equivalence relation and could be seen as the top of the most interesting semantics orders, and ending with *Trace semantics*, which is the simplest of the semantics that defines an order, my colleagues

Definition 7. *Let $\sqsubseteq$ be a behaviour preorder. Then a binary relation S over processes is a* bisimulation up-to $\sqsubseteq$*, if pSq implies that:*

- *For every a, if $p \xrightarrow{a} p'_a$, then there exist q' and q'_a, $q \sqsupseteq q' \xrightarrow{a} q'_a$ and $p'_a S q'_a$;*
- *For every a, if $q \xrightarrow{a} q'_a$, then there exist p' and p'_a, $p \sqsupseteq p' \xrightarrow{a} p'_a$ and $p'_a S q'_a$.*

Two processes are bisimilar up-to $\sqsubseteq$*, written $p \approx_{\sqsubseteq} q$, if there exists a bisimulation up-to $\sqsubseteq$, S, such that pSq.*

Figure 2.8: Definition of bisimulation up-to, as it appeared in [dFG05].

from Universidad Complutense de Madrid (UCM) looked for an adequate modification of the notion of bisimulation that could be used to define the equivalences induced by any coarser semantics. In this way, the concept of global bisimulation appeared. It can be defined by the bisimulation game, in which the defender can transform the transition system in some (limited) way before making his moves. Thereby, it has more possibilities of getting the equivalence between the compared processes, and as a consequence it is possible to characterize in this way some other relationship with less identification power than bisimulation.

Although (adequate) global bisimulations characterize the equivalences induced by each of the semantics in the *ltbt-spectrum*, the proofs needed to prove that characterization were different for each case. In order to get a general definition that would make possible generic proofs, a bigger abstraction was necessary. A first approximation was the concept of *bisimulation up-to preorder* presented in Fig. 2.8. This new idea of bisimulation used the preorder properties to avoid an specific use of the corresponding semantics.

In a similar way, it was possible to define the notion of *simulation up-to*, whose nice properties came from its coinductive characterization. However, there were still some pending questions in order to totally understand how everything works in a unified way. In particular, when working under simulation up-to, they still needed different proof techniques than those used for the case of bisimulation up-to. The final key to organize the *ltbt-spectrum* was given by the notion of *C-constrained simulation*, presented in Fig. 2.9.

An important milestone in the work by C. Gregorio-Rodríguez and my supervisor was to discover that the generalized notion of simulation was really the spinal bone of the different semantics in the *ltbt-spectrum*. With this intuition, a new way of looking to the *spectrum* appeared –See Fig. 2.7–. The new structure organizes the *ltbt-spectrum* in different layers. Each of them is governed by a *C-constrained simulation* which divides the layers according to three different kinds of observations. These observations are formally defined in Fig. 2.10 –go

Definition 8. Given a relation C over BCCSP processes, a relation S_C is a *C-constrained simulation*, if pS_Cq implies:

– For every a, if $p \xrightarrow{a} p'$ there exists q', $q \xrightarrow{a} q'$ and $p'S_Cq'$, and
– pCq.

We say that process p is C-simulated by process q, or that q C-simulates p, written $p \sqsubseteq_{\rightarrow}^{C} q$, whenever there exists a C-constrained simulation S_C, such that pS_Cq.

Figure 2.9: Definition of C-constrained simulation, as it appeared in [dFG08b].

to the paper in Section 5.2 for details. Since these details correspond to the part of that paper that is not formally a part of this thesis, please excuse us for not repeating them here–. However, we try to briefly explain below the role of each kind of observations in the structure of the new spectrum –Fig. 2.7 page 43–.

1. Branching observations (trees) which corresponds to the semantics at the left hand side of each of the layer, governed by a different *C-constrained* simulation (e.g. RS).

2. Linear observations (decorated traces) which define the semantics forming the "diamond" on the right of the layers. They constitute the classical denotational linear semantics (e.g. RT, R, FT, F).

3. Deterministic observations (deterministic trees) defining the semantics which are in the middle, between the branching and the linear semantics (e.g. PW). These are in fact "semi-branching" semantics.

In the first decade of the 21^{th} century some other notions of semantics appeared that were not considered by R. J. van Glabbeek in [vG01]. Among these, was the *Revivals semantics*, introduced by Bill Roscoe [RRS07, Ros09]. These new semantics also appeared when my supervisor David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino, from my group at *UCM*, re-structured the spectrum in [dFGP09a], where also some other new semantics, that remained without a name, emerged as a consequence of the systematic classification.

The different observations, and the concept of constrained simulation, have also been the key support in the development of our unified logical characterizations.

Definition 5. *1. A* branching general observation *(bgo for short) of a process is a finite, non-empty tree whose arcs are labeled with actions in Act and whose nodes are labeled with local observations from L_N, for N a constraint; the corresponding set BGO_N is recursively defined as:*
- $\langle l, \emptyset \rangle \in BGO_N$ *for* $l \in L_N$.
- $\langle l, \{(a_i, bgo_i) \mid i \in 1..n\} \rangle \in BGO_N$ *for every* $n \in \mathbb{N}$, $a_i \in Act$ *and* $bgo_i \in BGO_N$.

2. The set $BGO_N(p)$ of branching general observations of p corresponding to the constraint N is

$$BGO_N(p) = \{\langle L_N(p), S \rangle \mid S \subseteq \{(a, bgo) \mid bgo \in BGO_N(p'), p \xrightarrow{a} p'\}\}.$$

3. We write $p \leq_N^b q$ if $BGO_N(p) \subseteq BGO_N(q)$.

Definition 6. *1. The set LGO_N of* linear general observations *(lgo for short) for a local observer L_N is the subset of BGO_N defined as:*
- $\langle l, \emptyset \rangle \in LGO_N$ *for each* $l \in L_N$.
- $\langle l, \{(a, lgo)\} \rangle$, *whenever* $a \in A$ *and* $lgo \in LGO_N$.

2. The set of linear general observations *of a process p with respect to the local observer L_N is $LGO_N(p) = BGO_N(p) \cap LGO_N$.*

Definition 12. *1. We say that a bgo is* deterministic *if for every node in it, its set of children $\{(a_i, bgo_i)\}$ satisfies $a_i \neq a_j$ whenever $i \neq j$. We denote with $dBGO_N$ the set of deterministic observations in BGO_N.*

2. The set of deteministic branching observations (dbgo for short) of a process p is $dBGO_N(p) = BGO_N(p) \cap dBGO_N$.

3. We write $p \leq_N^{db} q$ if $dBGO_N(p) \subseteq dBGO_N(q)$.

Figure 2.10: Resp. branching, linear and deterministic observations in [dFGP09b].

2.3.2 Van Glabbeek's logical characterizations of the semantics

Even if C. Gregorio-Rodríguez together with my supervisor, and now myself with the latter, have provided a much more structured *spectrum*, it is clear that our "Table of Elements" would not have been possible without the previous work done by R. J. van Glabbeek in [vG01]. There he collected (and personally developed in some of the cases) the characterizations of the different semantics that had been introduced by several authors along the years. In particular, considering the logical approach, R.J. van Glabbeek looked for sets of logical formulas as simple (and hence as small) as possible, characterizing each semantics. Probably he was driven by the idea that a small set of formulas would make simpler any further study on the characterized semantics. As a consequence of that decision, his collection of logical characterizations did not follow any common pattern.

Table 2.1 contains the characterization of the semantics in van Glabbeek's spectrum as it was presented in [vG01]. $\mathcal{L}_Z$ with $Z \in \{T, CT, F, FT, R, RT, PF, S,$

Formulas \ Semantics ($\mathcal{Z}$)	T	S	CT	CS	F	FT	R	RT	PW	RS	PF	2S	B
$\top \in \mathcal{L}_\mathcal{Z}$	•	ν	•	ν	•	•	•	•	ν	ν	ν	ν	ν
$\mathbf{0} \in \mathcal{L}_\mathcal{Z}$			•	•	ν	ν	ν	ν	ν	ν	ν	ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z}$, $a \in Act \Rightarrow$ $a\varphi \in \mathcal{L}_\mathcal{Z}$	•	•	•	•	•	•	•	•	ν	•	•	•	•
$X \subseteq Act \Rightarrow$ $\widetilde{X} \in \mathcal{L}_\mathcal{Z}$					•	ν	ν	ν	ν	ν	ν	ν	ν
$X \subseteq Act \Rightarrow$ $X \in \mathcal{L}_\mathcal{Z}$							•	ν	•	•	ν	ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z}$, $X \subseteq Act \Rightarrow$ $\widetilde{X}\varphi \in \mathcal{L}_\mathcal{Z}$						•		ν	ν	ν		ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z}$, $X \subseteq Act \Rightarrow$ $X\varphi \in \mathcal{L}_\mathcal{Z}$								•	ν	ν		ν	ν
$\varphi_i \in \mathcal{L}_\mathcal{Z}\ \forall i \in I \Rightarrow$ $\bigwedge_{i\in I} \varphi_i \in \mathcal{L}_\mathcal{Z}$		•		•						•		•	•
$X \subseteq Act$, $\varphi_a \in \mathcal{L}_{PW}\ \forall a \in X \Rightarrow$ $\bigwedge_{a\in X} a\varphi_a \in \mathcal{L}_\mathcal{Z}$									•	ν		ν	ν
$\varphi_i, \varphi_j \in \mathcal{L}_T\ \forall i \in I \forall j \in J \Rightarrow$ $\bigwedge_{i\in I} \varphi_i \wedge \bigwedge_{j\in J} \neg\varphi_j \in \mathcal{L}_\mathcal{Z}$											•	ν	ν
$\varphi \in \mathcal{L}_S \Rightarrow$ $\neg\varphi \in \mathcal{L}_\mathcal{Z}$												•	ν
$\varphi \in \mathcal{L}_\mathcal{Z} \Rightarrow$ $\neg\varphi \in \mathcal{L}_\mathcal{Z}$													•

Table 2.1: Van Glabbeek's logical characterizations for the semantics.

$CS, RS, 2S, PW, B\}$ denotes each of the logics; the dots indicate the clauses that we need to introduce to obtain the corresponding logical languages; and the boxes marked with ν correspond to rules that could be added to $\mathcal{L}_Z$, but would only introduce redundant formulas. The following connectives, which appear in the table, are not in $\mathcal{L}_{HM}$ (See Def. 4 in page 18), but can be obtained as syntactic sugar:

$$\widetilde{X} := \bigwedge_{a\in X} \neg a\top \qquad \widetilde{X}\varphi' := \widetilde{X} \wedge \varphi' \qquad 0 := \widetilde{Act}$$

$$\varphi_1 \wedge \varphi_2 := \bigwedge_{i\in\{1,2\}} \varphi_i \qquad X := \bigwedge_{a\in X} a\top \wedge \bigwedge_{a\notin X} \neg a\top \qquad X\varphi' := X \wedge \varphi' \qquad \widetilde{a} := \neg a\top$$

In our work we followed the opposite approach of that chosen by R. J. van Glabbeek. For each semantics defined by a preorder, we have found a (larger) language $\mathcal{L} \subseteq HML$ characterizing it, which provides the unified characterization presented in [RdF11, dFGPR13].

Again, as a consequence of the systematic approach, we contributed to the appearance of some other new semantics in [RdF11] when we studied the unification of the logical characterization of the semantics in the re-structured spectrum [dFGP09a]. It was not until we developed the unification of the logical semantics

that the seed for these never considered before linear semantics appeared. See Chapter 5 for more details.

2.4 Semantics and measures

In the last years, there has been a growing interest in the relationships between semantics and measures. The published papers are mostly interested in the definition of distances between processes whose behavior differs up to a certain value. Of course, the first thing to be stated is how do we exactly fix those behaviors, that is, with respect to which semantics are we defining our distance?. In order to do that, they start by quantifying how different the behavior of one process (specification) is from another (implementation), by means of the so called *(Bi)simulation Distance Game* [dAFS09, CHR10, FLT11, FL14]. This game is defined in the same way as the (bi)simulation game –See Def. 9 in page 35–, but in this case, we allow a process to "cheat" when (bi)simulating the move of the other. However, such mismatched transitions are penalized according to an error model. Then, the (bi)simulation distance between two processes, with respect to this error model, is the lowest penalty achievable by the Defender when playing the quantitative (bi)simulation game starting with the two processes.

Fig. 2.11 presents the formal definition of the quantitative simulation game.

Given two weighted transition systems A and A' with the same alphabet, we define the corresponding *quantitative simulation game graph* $G_{A,A'} = \langle S^G, S_1^G, S_2^G, \Sigma, E^G, s_0^G \rangle$ as follows:

1. The alphabet Σ is the same as the alphabet of A and A'.
2. The state space $S^G = S \times (\Sigma \cup \{\#\}) \times S' \cup \{s_{\text{err}}\}$.
3. The states are partitioned into Player 1 and Player 2 states as follows: $(s, \#, s') \in S_1^G$, and $(s, \sigma, s') \in S_2^G$ for all $\sigma \in \Sigma$. Also, state $s_{\text{err}} \in S_1^G$.
4. The initial state is $s_0 = (s_0, \#, s_0')$.
5. Each transition of the game graph corresponds to a transition in either A or A' as follows:
   (a) $((s, \#, s'), \sigma, (t, \sigma, s')) \in E^G \Leftrightarrow (s, \sigma, t) \in E$
   (b) $((s, \sigma, s'), \sigma, (s, \#, t')) \in E^G \Leftrightarrow (s', \sigma, t') \in E'$
   For each of the above transitions, the weight is the same as the weight of the corresponding transition in A or A'.
6. If there is no outgoing transition from a particular state, transitions to s_{err} are added with all symbols. s_{err} is a sink with transitions to itself on all symbols. Each of these transitions has weight 1.

Figure 2.11: The Quantitative Simulation Game as presented in [CHR10].

Thus, we have the definition of the *Quantitative Simulation Game* where the

objective (or strategy) of the Attacker is to maximize the value of the play, while the Defender tries to minimize it. [CHR10] also shows that, by using several variations of the simulation game, i.e. changing the moves allowed to each player, it is possible to measure several different properties (such as correctness, coverage, and robustness) of the implementation with respect to the specification.

- **Correctness Distance.** Roughly speaking it measures the degree of incorrectness that an implementation contains with respect to an specification. Therefore, it looks for the simulation of the system I by S. As a consequence, the Attacker plays on I trying to force the specification S to make as many errors as possible when replying it, while the Defender tries to show that the number of errors in I is as small as possible. An error is just the execution of some action instead of the required one at the time. Of course, if the implementation is correct (that is, S simulates I) then the correctness distance will be 0.

- **Coverage Distance.** It is the dual game of the game presented above. It measures the behaviors that are in one system, but not in the other. In this case, for a specification S and an implementation I, the coverage distance corresponds to the behavior of the specification which is the farthest from any behavior of the implementation. The game gives the distance from I to S as the minimal number of errors that have to be committed by I in order to cover all the behaviors of S.

- **Robustness Distance.** It is the measure of the number of errors that the implementation has to make in such a way that it does not conform with the specification. In other words, it gives us the (minimal) number of critical points where simultaneous errors will lead to an unacceptable behavior. This means that we get another dual to the correctness distance above, but in a total different way. As a matter of fact, in this case the value of the distance means *how close are the implementation and the specification.* Now, the Attacker plays on I committing errors that the Defender has to reply on S. Thus, the former tries to finish the game as soon as possible, while the latter tries to remain in the play as much as possible.

Independently, U. Fahrenberg and A. Legay presented in *The quantitative linear-time-branching-time spectrum* [FL14] a distance-agnostic approach to quantitative verification, based on the *Quantitative Ehrenfeucht-Fraïssé game* [Ehr61, EM79]. This game followed the same ideas presented above for the case of the simulation game. As a novelty, they modified the strategies enabled to each player in order to obtain a definition of distance not only for the branching semantics

(bisimulation, simulation, ready simulation, k-nested simulation semantics), but also for the linear ones (readies and trace semantics). For that purpose, they used a *switch counter*, which allowed the attacker to change a limited number of times the process where he played along the game executions. Also they included *blind strategies*, where the choices of the Attacker did not depend on those of the Defender. In Fig. 2.12 we present the notion of blind strategy defined by U. Fahrenberg and A. Legay.

any subset $\Theta_1' \subseteq \Theta_1$ we define the set of blind Θ_1'-strategies by

$$\tilde{\Theta}_1' = \{\theta_1 \in \Theta_1' \mid \forall \pi, \rho, \rho', m : \theta_1(\pi, \rho, m) = \theta_1(\pi, \rho', m), \\ \text{or } \theta_1(\pi, \rho, m) = (e, m+1) \text{ and } tgt(\text{last}(\rho)) \neq tgt(\text{last}(\rho'))\}.$$

Hence in such a blind strategy, either the edge chosen by Player 1 does not depend on the choices of Player 2, or the switch counter is increased, in which case the Player-1 choice only depends on the target of the last choice of Player 2 (note that this dependency is necessary if Player 1 wants to switch paths).

Figure 2.12: The blind strategy in [FL14].

These two authors also define a notion of distances for *Modal Transition Systems* [FL14], by introducing a general framework of quantitative refinement for quantitative specifications, together with a natural notion of quantitative relaxation of specifications. Another example dealing with the definition of distances between linear semantics can be found in [dAFS09]. There are other approximations that also compute distances between processes. This is the case of the work by K. G. Larsen, U. Fahrenberg and C. R. Thrane [TFL10, LFT11], where they define the *Weighted Transition Systems* for the analysis of quantitative and qualitative properties of reactive systems and study the point-wise and the accumulating simulation distances. P. Panangaden and J. Desharnais have several works on distances within the framework of *Markov Processes*, see for instance [DGJP99, DGJP04]. L. de Alfaro et al. have been working with probabilistic systems [dAMRS08], and F. van Breugel and J. Worrell with probabilistic transition systems [vBW05, vBSW08]. For the latter, the main idea was to find an adequate logic language that captures the quantitative behavior of the systems, in the sense that for each formula φ and each system s, we have a value $[\varphi](s)$ that gives us the *level of satisfaction of* φ *in* s (Def. 3.2 in Fig. 2.13). Then, we define the distance between two systems as that shown by the difference between the values provided by the formula that distinguishes them the most –See Def. 3.4 in Fig. 2.13–. The curious thing about this definition is that the level of satisfaction of a formula does not necessarily correspond to the probability with which it is satisfied, as one could expect. By the way, it seems no easy at all to find a clear intuitive meaning for these values in isolation: they only get a (relative) meaning when comparing two systems.

Definition 3.2. Given a probabilistic transition system $\langle S, \pi \rangle$ and a discount factor $\delta \in (0,1]$, for each $\varphi \in \mathcal{L}$, the function $[\![\varphi]\!]_\delta : S \to [0,1]$ is defined by

$$\begin{array}{rcl} [\![\text{true}]\!]_\delta(s) & = & 1 \\ [\![\Diamond\varphi]\!]_\delta(s) & = & \delta \sum_{s' \in S} \pi(s,s')[\![\varphi]\!]_\delta(s') \\ [\![\varphi \wedge \psi]\!]_\delta(s) & = & \min\{[\![\varphi]\!]_\delta(s), [\![\psi]\!]_\delta(s)\} \\ [\![\neg\varphi]\!]_\delta(s) & = & 1 - [\![\varphi]\!]_\delta(s) \\ [\![\varphi \ominus q]\!]_\delta(s) & = & \max\{[\![\varphi]\!]_\delta(s) - q, 0\} \end{array}$$

Definition 3.4. Given a probabilistic transition system $\langle S, \pi \rangle$ and a discount factor $\delta \in (0,1]$, the distance function $d_\delta : S \times S \to [0,1]$ is defined by

$$d_\delta(s_1, s_2) = \sup_{\varphi \in \mathcal{L}} [\![\varphi]\!]_\delta(s_1) - [\![\varphi]\!]_\delta(s_2).$$

Figure 2.13: Definition of a distance for probabilistic systems in [vBSW08].

We have contributed to the field of distances between process semantics by proposing an alternative approach –See Chapters 6, 7– which considers a global view. Our distance takes into account how the set of all possible behaviors of one process differs from the set of possible behaviors of the other, and not just the maximum disagreement, that is what the different variations of the quantitative (bi)simulation game can do. We could also define some variants of our distance capturing robustness, coverage, etc, but this remains as future work.

Chapter 3

Our contribution to the scientific community

"... you must keep in view what you are, striving to know yourself, the most difficult thing to know that the mind can imagine."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter-42 Part-2

"... has de poner los ojos en quien eres, procurando conocerte a ti mismo, que es el más difícil conocimiento que puede imaginarse."

This chapter is devoted to present the main contributions produced by this thesis. As we previously said in Section 1.5, our trip around the world of formal methods and process semantics started with my master's thesis, titled *Caracterizaciones lógicas uniformes de las semánticas de procesos* [Rom10].

We will describe our work during these years as an exploration voyage. In this tour, we have visited many different places, some of them beautiful, in the sense that they have produced some of our new results; some of them interesting, where we have discovered unexpected problems, which provided new paths to explore, several of which we have already walked, and some others left for the future; or too exotic and perhaps even dangerous or too abrupt, which led us to non exit paths (at least for the moment). However, as a good explorer, from each one of these places we always picked up beautiful memories. Even the most difficult steps meant, for sure, good experiences that facilitated me to grow up as a researcher. In fact, although we expect that the work collected in this thesis will be worth the trouble, it is also true that we decided to conclude the story with some intrigue –see Section 4.2 for details– that has given us more than a headache. Moreover, now there are some interesting directions to follow, and although we took a break in order to write this thesis, we will continue our trip in a short time with more work that hopefully will produce some new results.

After exploring the field, the first direction that we followed was to complete the work on the unification of the semantics in the *ltbt-spectum*, by revisiting the work by R. J. van Glabbeek [vG01]. My supervisor, D. de Frutos Escrig, started this work mainly in C. Gregorio Rodríguez's Ph.D. thesis [GR09]. However, due to the complexity and volume of their work, initially they only developed the operational and the axiomatic unified frameworks, but gave the logical framework up. We have developed the logical part of the unified *ltbt-spectrum* in our work [RdF11, dFGPR13] – see the publications in Chapter 5–. It was interesting to note how the logical and the observational framework [dFGP09b] are strongly related. Furthermore, the duality between the axiomatic and the logical framework empowered us by discovering two new semantics for each layer in the *spectrum*, that had remained unnoticed during the previous unification work. Now, with the logical unification, they appeared almost in an immediate way. Certainly, the weak semantics [vG93] still remain as a quite interesting challenge. However, the role of non-determinism, with its apparent simplicity but multiple embarrassing faces, makes the work extremely complicated, at least, as far as all the possibilities are intended to be covered in a single framework.

Once we concluded the logical aspects of the semantics which constitute our objective 1 (see Section 1.5.1, page 23), an interesting path to follow appeared. We had been able to classify the semantics, and thus the equivalences induced by

them, but *What happens with those processes that are not equivalent under a given semantics?*. The same question had been recently posed by other authors, and some of them proposed a definition of distance based on the bisimulation game –see for instance [CD08, CHR10]–. With these definitions in mind we tackled the problem of defining some more adequate distance between processes. A first approximation dealt with finite processes [RdF12a, RdF12b] (see the publications in Chapter 6), where we soon discovered that the previously proposed notions did not satisfy our expectations. We started to explore in a new direction, which led us to our objective 2.1 (see Section 1.5.1, page 24) and later to 2.2 (see Section 1.5.1, page 24).

The last part of our research, [RdF14, RdFM15] (see the publications in Chapter 7), has partially been developed during my three stays of one month each at Reykjavík University. There, under the supervision of Luca Aceto, I had the opportunity to collaborate with the other members of his group involved in our *NILS project*: Anna Ingólfsdóttir, Ignacio Fábregas, and specially Dario Della Monica. The latter also enjoyed a three months long stay in Madrid, along which we started a collaboration that has produced at the moment as main result the last publication included in this thesis.

After this brief summary describing our research during the last five years, in the following we will explain in more detail the most important results of the six publications that constitute the main body of this thesis. In order to do it more readable, we will classify the results according to the objectives presented in Section 1.5.1. More precisely, in *"A logical unification for the semantics"* (Section 3.1), we accomplish the completion of our objective 1; while in *"Defining distance between processes"* (Section 3.2), we will tackle our objective 2.1. Finally, in *"Defining distances between infinite processes"* (Section 3.3), we will explain all the work related with objective 2.2.

The definitions, theorems and figures that appear in the next sections have been extracted directly from the original papers where they are included.

3.1 A logical unification for the semantics

The previous work about the unification of the *ltbt-spectrum* [GR09] had ordered in a systematic way the observational (or testing) and axiomatic characterizations of the semantics in the original *spectrum* [vG01]. There, R. J. van Glabbeek also gave logical characterizations $\mathcal{L}_s$ for each of the semantics in his *spectrum*, but again these characterizations were obtained in a quite *ad hoc* way, and hence it was difficult to relate these semantics starting from their characterizations. So, it was not clear to establish the main properties that each semantics captures.

The formulas of the *Hennessy-Milner Logic* had a too low level structure induced by the combination of prefix and conjunction. We looked for a higher level structure where the role of both kinds of operators would become clearer. This, joined with the differences between branching-time and linear-time semantics, was the key to produce the different semantics. Using the fact that each language $\mathcal{L} \subseteq HML$ defines a preorder $\prec_{\mathcal{L}}$, given by $p \prec_{\mathcal{L}} q \Leftrightarrow (p \models \varphi \Rightarrow q \models \varphi)$, we opted for the dual approach to the one followed by R. J. van Glabbeek. Therefore, we looked for a set of formulas as large as possible, characterizing each semantics in a natural way, instead of looking for a characterizing logical language as small as possible (as it seemed to be the intention of van Glabbeek in his work [vG01]). In this way, we were able to produce in [RdF11, dFGPR13] all the logical characterizations of the semantics in a uniform way, by relying on the previous work [GR09]. The key point was to use negation and conjunction in a properly tuned way.

It was particularly interesting to obtain, as a result, that whenever a semantics is finer than other, then our logical characterization of the former simply contains that of the latter (making that relationship immediate). In general, this is not always the case: we could have S and S' logically characterized by $\mathcal{L}_S$ and $\mathcal{L}_{S'}$ respectively, with S' finer than S, but $\mathcal{L}_S \not\subseteq \mathcal{L}_{S'}$. We can generate some dummy counterexamples, simply adding to $\mathcal{L}_S$ any redundant formula not in $\mathcal{L}_{S'}$ that does not change the equivalence characterized by $\mathcal{L}_S$. But, some much more interesting counterexamples can be obtained looking at van Glabbeek's characterizations. For instance, you can easily check that $\mathcal{L}_F$ in Table 2.1 is not contained at all in $\mathcal{L}_R$, and this makes no trivial to check that readiness is finer than failures starting from these logical characterizations.

We also divided the semantics into the linear ones and the branching ones, as done in [vG01], but the main idea used in the development of the logical unification framework, was again the concept of *C-constrained simulation* defined by C. Gregorio Rodríguez [dFG08a] –See Fig. 2.9 in page 45–. Thus, each layer in the spectrum is governed by a branching semantics, given by an N-constrained simulation.

Our research on the logical framework started with my master's thesis [Rom10], and in 2011 we were able to produce a nice presentation of these results in order to transmit them to the scientific community in an adequate forum. We presented our logical characterization in [RdF11]. It can be briefly summarized as follows: apart from bisimulation, the other equivalences can be arranged in two dimensions. The first axis has three possible values: Simulations, Deterministic Branching Semantics and Linear Semantics. On the other axis, we have the local observations (or constraints): Universal (i.e., nothing is observed on the states),

Complete (i.e., we observe when the processes cannot move), Ready (i.e., we observe the set of offered actions), Trace, and Simulation.

Fig. 3.1 shows the logical characterization of each of the constraints that are used in the (unified) definition of the most popular semantics. From the simplest one, *the Universal semantics (U)*, which only contains the formula $\top$, and is the restriction used to characterize the *simulation semantics*; to the highest (letting alone bisimulation): the *Simulation itself (S)*, which contains every formula in *HML* but negation, and enables us to generate the *k-nested simulations*. In the table we mark with a dot ($\bullet$) the formulas that are needed to define each of the logical languages, while we use ν to indicate those formulas that can be included in the logic, but are not really needed in the characterizing logical language, because they are redundant.

Formulas \ Constraints ($\mathcal{N}$)	U	C	I	T	S	B
$\top \in \mathcal{L}'_{\mathcal{N}}$	$\bullet$	$\bullet$	$\bullet$	$\bullet$	ν	ν
$\neg\top = \bot \in \mathcal{L}'_{\mathcal{N}}$	ν	ν	ν	ν	ν	ν
$\neg 0 \in \mathcal{L}'_{\mathcal{N}}$		$\bullet$	$\bullet$	ν	ν	ν
$a \in Act \Rightarrow a\top \in \mathcal{L}'_{\mathcal{N}}$			$\bullet$	ν	ν	ν
$\varphi \in \mathcal{L}'_{\mathcal{N}}, a \in Act \Rightarrow a\varphi \in \mathcal{L}'_{\mathcal{N}}$				$\bullet$	$\bullet$	$\bullet$
$\varphi_i \in \mathcal{L}'_{\mathcal{N}}\ \forall i \in I \Rightarrow \bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{\mathcal{N}}$					$\bullet$	$\bullet$
$\varphi \in \mathcal{L}'_{\mathcal{N}} \Rightarrow \neg\varphi \in \mathcal{L}'_{\mathcal{N}}$						$\bullet$

Figure 3.1: N-Constraints for semantics as presented in [RdF11].

As we saw in detail in Section 2.3, the linear semantics produced a "diamond" –See [dFGP09a]– because they were not linearly ordered, due to the observation of *readiness* or/and *failure traces*. It is interesting to observe that, until we use the *Initial semantics (I)* as a constraint, we do not have a real diamond in the corresponding layer of the *spectrum*. This is caused by the simplicity of the corresponding constraint, which makes that all the linear semantics in the first two layers collapse into the *Trace semantics (T)*, produced by the *Universal semantics*, and the *Complete Trace semantics (CT)*, generated by the constraint given by the *Complete semantics (C)*.

The rules to obtain the different logical characterizations of the semantics, using conjunction and negation in an adequate way, can be roughly defined as follows:

– The branching semantics (that is the corresponding *N-constraint simula-*

tion) has an unrestricted use of the conjunction and negation operators. This means that, whenever we have a formula belonging to the corresponding language, $\mathcal{L}_{NS}$, we can simply deny it or combine it with any other, using even arbitrary conjunctions.

– Concerning the linear semantics, which take into account the positive information (that is *readiness*- and *ready-traces*-like semantics), we use our *Symmetric closure* $\mathcal{L}_N^{\equiv}$. In this case, we need to exactly capture which actions are allowed. In order to do that, we need to list separately at each state, both the actions that we can do and those that we cannot.

– With respect to the linear semantics which only take into account the negative information (that is *failures*- and *failure-traces*- like semantics), we use our *Negative closure* $\mathcal{L}_N^{\neg}$. In this case, it is enough to know which actions cannot be performed at each time (without worrying about the actions we can do).

We refer the reader to our paper [RdF11] included in this thesis (see Section 5.1) in order to get the formal definition of our symmetric and negative closures. Next, the point is how to distinguish the semantics which consider the positive (resp. negative) information; that is, how we can distinguish between *readiness* and *ready trace semantics* (resp. *failure* and *failure trace*). Again, by restricting the use of conjunction, we can achieve it. For the first case, which corresponds to the semantics that does more identifications, we do not allow the use of conjunctions between formulas in $\mathcal{L}_R$ (resp. $\mathcal{L}_F$). Instead, the case of *ready trace* (resp. *failure trace*) semantics allows for mixing a formula in $\mathcal{L}_R$ (resp. $\mathcal{L}_F$) with a trace in $\mathcal{L}_N^{\equiv}$ (resp. $\mathcal{L}_N^{\neg}$).

We present in Fig. 3.2 our logical characterizations of the semantics for each layer in the spectrum. In the second row we have the semantics corresponding to the particular case using the *Initial semantics (I)* as constraint. This constraint gives us the layer of the *Ready Simulation semantics (RS)*, for sure the most important one in the *unified ltbt-spectrum*.

Now, with the unified characterization of the semantics, we have a clearer picture of the spectrum. We can use the parameterized definition provided in [RdF11] to prove generic properties of all (or a part of) the semantics in a generic way, without having to repeat similar proofs. Furthermore, our unified logical characterization capacitated us to discover two new linear semantics in each layer of the spectrum. Following the ideas about *symmetric and negative closure*, we immediately thought about another variant: the duality of *negative closure*, i.e., simply considering the actions that we can do (and not those that we cannot do). As in the previous case, these considerations offer us two new possibilities:

Formulas \ Semantics ($\mathcal{Y}_N$)	$\preceq_N^{\mathbf{If}\supseteq}$	$\preceq_N^{\mathbf{If}}$	$\preceq_N^{\mathbf{I}\supseteq}$	$\preceq_N^{\mathbf{I}}$	D_N	NS	$N \in \{U,C,I,T,S\}$
	F	R	FT	RT	PW	RS	when $N = I$
$\top \in \mathcal{L}'_{\mathcal{Y}_N}$	•	•	•	•	•	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_N},\ a \in Act \Rightarrow$ $a\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$	•	•	•	•	ν	•	
$\varphi \in \mathcal{L}_N^{\neg} \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$	•	ν	ν	ν	ν	ν	
$\varphi \in \mathcal{L}_N^{=} \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$		•		ν	ν	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_N},\ \sigma \in \mathcal{L}_N^{\neg} \Rightarrow$ $\sigma \wedge \varphi \in \mathcal{L}'_{\mathcal{Y}_N}$			•	ν	ν	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_N},\ \sigma \in \mathcal{L}_N^{=} \Rightarrow$ $\sigma \wedge \varphi \in \mathcal{L}'_{\mathcal{Y}_N}$				•	•	ν	
$X \subseteq Act,\ \varphi_a \in \mathcal{L}'_{\mathcal{Y}_N}\ \forall a \in X \Rightarrow$ $\bigwedge_{a\in X} a\varphi_a \in \mathcal{L}'_{\mathcal{Y}_N}$					•	ν	
$\varphi_i \in \mathcal{L}'_{\mathcal{Y}_N}\ \forall i \in I \Rightarrow$ $\bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{\mathcal{Y}_N}$						•	
$\varphi \in \mathcal{L}_N \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$						•	
$\varphi \in \mathcal{L}_N \Rightarrow$ $\neg\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$						•	

Figure 3.2: Our logical characterization, as it was presented in [RdF11].

depending on whether we consider the conjunction between a trace in $\mathcal{L}_N^{\surd}$ and a formula producing the *Partial Offer Trace semantics (POT)*, or not. The latter defines what we call *Partial Offer semantics (PO)*.

Definition 15 1. *The semantics of* ***partial offer traces*** *for the constraint N is that defined by the logic* $\mathcal{L}'_{\preceq_N^{\mathbf{I}\subseteq}}$ *with* $\top \in \mathcal{L}'_{\preceq_N^{\mathbf{I}\subseteq}};\ \varphi \in \mathcal{L}'_{\preceq_N^{\mathbf{I}\subseteq}},\ \sigma \in \mathcal{L}_N^{\surd} \Rightarrow \sigma \wedge \varphi \in \mathcal{L}'_{\preceq_N^{\mathbf{I}\subseteq}};\ \varphi \in \mathcal{L}'_{\preceq_N^{\mathbf{I}\subseteq}},\ a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\preceq_N^{\mathbf{I}\subseteq}}$.

2. *The semantics of* ***partial offers*** *for the constraint N is that defined by the logic* $\mathcal{L}'_{\preceq_N^{If\subseteq}}$ *with* $\top \in \mathcal{L}'_{\preceq_N^{If\subseteq}};$ $\sigma \in \mathcal{L}_N^{\surd} \Rightarrow \sigma \in \mathcal{L}'_{\preceq_N^{If\subseteq}};\ \varphi \in \mathcal{L}'_{\preceq_N^{If\subseteq}},\ a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\preceq_N^{If\subseteq}}$.

Duality between failures and partial offers causes the picture of the complete layer of linear semantics for each N to become two diamonds that share the side corresponding to the readies-based semantics.

Figure 3.3: New semantics discovered as they were presented in [RdF11].

By taking into account these facts, the natural presentation of the whole *spectrum* should not only contain a diamond, but a more elaborate structure: *a double diamond* sharing the side corresponding to the *readies-based* semantics, as presented in Fig. 3.4.

Also, our study enabled us to find an unexpected (but interesting) fact: *the logical characterization of Possible World semantics (PW) proposed by van Glabbeek was wrong*, thus proving our motto asserting that many times a general approach

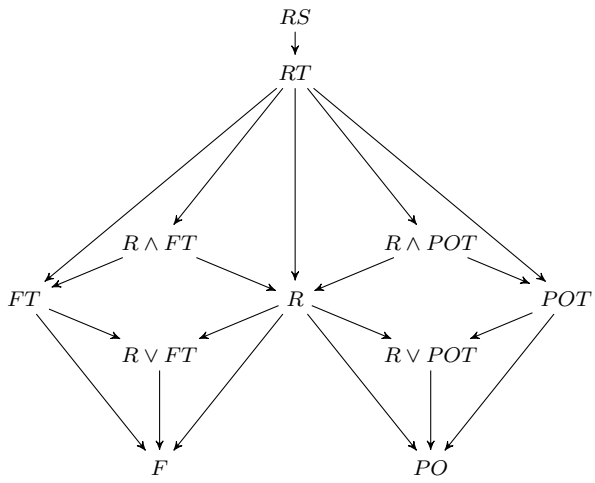


Figure 3.4: The couple of double diamonds in the "full" spectrum below RS.

to a problem produces more and better results than an *ad hoc* attack that is concentrated on the specific characteristics of the (concrete) problem. Roughly speaking, the possible worlds of a process are defined by the complete deterministic subtrees of their full tree of computations. Here deterministic means that no node has two successors labeled by the same action, while complete means that any internal node has successors labeled by all the actions in its initial set –See [vG01, dFGPR13] for more details–.

The processes in Fig. 3.5 serve as a counterexample showing that the PW's logic in [vG01] was wrong. Since that logic cannot "observe" intermediate offers, it cannot detect that the possible world $\{abc\}$ in p is different from all the possible worlds in q. On the contrary, our definition of PW empowers us by distinguishing these two processes. The formula $\varphi \equiv a(\neg d \wedge bc)$ is satisfied by p, but not by q. Thus, we easily proved once more time, that the simpler solution was the good one.

We have commented before that after our study of the logical framework, we collaborated with C. Gregorio Rodríguez and M. Palomino to arrange the whole unification theory [dFGPR13]. This paper essentially collected the work on the topic included in C. Gregorio Rodríguez's Ph.D. Thesis [GR09], and our work on the logical framework. We also established new connections between all the studied frameworks, hence the characteristics of all of the semantics in the spectrum appear in a clearer way. Certainly, only the sections regarding the logical framework from this encyclopedic paper are our novel contributions. In

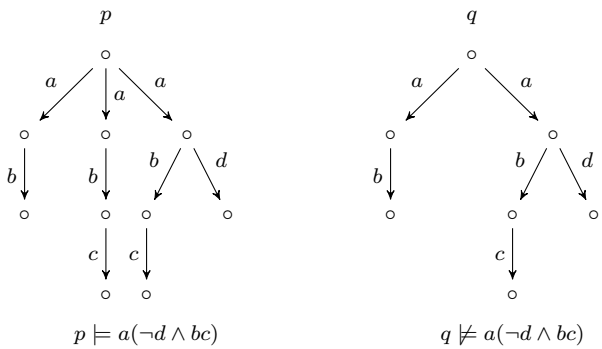


Figure 3.5: Trees showing the wrong characterization of PW.

particular, in Section 7 of the paper, we presented the theorems asserting the connections between the observational and the logical framework. We recall them in Fig. 3.6.

Theorem 7.8. *For each $N \in \{U, C, I, T, S\}$ and each $\mathcal{Y}_{\mathcal{N}} \in \{NS, \leq_N^l, \leq_N^{l\supseteq}, \leq_N^{lf}, \leq_N^{lf\supseteq}, \leq_N^{l\subseteq}, \leq_N^{lf\subseteq}, D_N\}$ in the spectrum there exists a correspondence between the set of complete normal formulas $\mathcal{CN}_{\mathcal{Y}_{\mathcal{N}}}(\mathcal{L}''_{\mathcal{N}})$ and the corresponding domain of observations ΩGO_N with $\Omega \in \{B, L\}$. This correspondence $\leftrightarrow$ satisfies that $\varphi \leftrightarrow \theta$ implies that $(p \models \varphi$ iff $\theta \in \Omega GO_N(p))$. Moreover:*

(1) *The set of complete normal formulas $\mathcal{CN}_{\mathcal{NS}}(\mathcal{L}''_{\mathcal{N}})$ (resp. $\mathcal{CN}_{\mathcal{D}_{\mathcal{N}}}(\mathcal{L}''_{\mathcal{N}})$) and the domain of branching general observations BGO_N (resp. $dBGO_N$) are isomorphic, that is, $\leftrightarrow$ is one to one.*

(2) *The set of complete normal formulas $\mathcal{CN}_{\leq_N^l}(\mathcal{L}''_{\mathcal{N}})$, $\mathcal{CN}_{\leq_N^{l\supseteq}}(\mathcal{L}''_{\mathcal{N}})$ and the domain of linear general observations LGO_N are isomorphic, that is, $\leftrightarrow$ is one to one.*

(3) *The set of complete normal formulas $\mathcal{CN}_{\leq_N^{lf}}(\mathcal{L}''_{\mathcal{N}})$ (resp. $\mathcal{CN}_{\leq_N^{lf\supseteq}}(\mathcal{L}''_{\mathcal{N}})$) and the quotient domain $LGO_N/_{\simeq_N^{lf}}$ (resp. $LGO_N/_{\simeq_N^{lf\supseteq}}$) are isomorphic, that is, $\leftrightarrow^{-1}$ is injective and $\varphi \leftrightarrow \theta$ iff $\theta \simeq_N^{lf\supseteq} \theta_\varphi$, for some adequate θ_φ.*

Theorem 7.10. *For each $N \in \{U, C, I, T, S\}$ and each $\mathcal{Y}_{\mathcal{N}} \in \{NS, \leq_N^l, \leq_N^{l\supseteq}, \leq_N^{lf}, \leq_N^{lf\supseteq}, \leq_N^{l\subseteq}, \leq_N^{lf\subseteq}, D_N\}$ in the spectrum, if we restrict ourselves to image-finite processes, the logical semantics $\sqsubseteq^f_{\mathcal{Y}_{\mathcal{N}}}$ induced by the logic $\mathcal{L}^f_{\mathcal{Y}_{\mathcal{N}}}$, is equivalent to the corresponding observational semantics in Definitions 4.2, 4.11 and 4.32. In order to unify our notation, here we will denote by GO_N the corresponding semantic domain.*

Figure 3.6: Equivalence between the observational and the logical framework [dFGPR13].

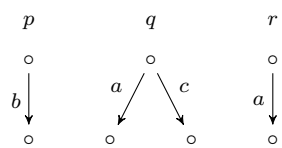
Now with our paper [dFGPR13], the previous commented duality between the axiomatic and the logical framework has become totally evident.

3.2 Defining distances between processes

Concerning the distances between processes semantics, we started by studying the wide collection of papers by Thomas A. Henzinger et al. [dAHM03, CHR10, CHR12, CCHR14] as well as into the papers written by U. Fahrenberg et al. [TFL10, LFT11, FTL11, FL13, FL14]. These papers used *the weighted bisimulation game* to define the desired distances as the values of these games. Their optimal strategies looked for the (individual) plays that produce the bigger difference. However, we noticed that in this way it is not possible at all to add the differences captured by each of the plays of the game, even when they are induced by totally independent *parts* of the compared processes. This is more or less the situation when one considers the usual distance in $\mathbb{R}^2$: the projected distance between two points wrt the two dimensions are not totally added. But certainly the distance between $(1, 1)$ and $(2, 2)$ is far from being only 1.

In this sense, we considered more appropriate to define a new notion of distance that takes into account all the differences at the same time, and not just the largest of them. Hence, our contribution presented along this section, dealt with the definition of that new distance for processes.

Certainly, bisimulation characterizes trees up to idempotency, and this works because a single disagreement between two trees (out of those avoidable by applying commutativity and idempotency) produces non equality. The *weighted bisimulation game* looks for the maximal disagreement between the branches of the trees, even if compared with the big strength of alternation, by using the *max* operator [dAMRS07]. As a consequence, additional independent disagreements are not taken into account. Therefore, when we find k independent disagreements, each of them of "size" one, the weighted bisimulation game still gives us 1 as distance, instead of the total disagreement, that should be k, in our opinion. We saw that the natural way of (formally) capturing the bound of total distance, was to add the cost of all the changes that we need to do in order to go from one of the compared processes to the other. We start by presenting a simple example to show why the previous framework was not adequate to "count" the number of disagreements, as explained above.



As detailed in [RdF12a], we looked for a notion of distance between processes that enabled us to distinguish the distance between p and q of that from p and

r. Let us consider the distance between actions induced by the lexicographic order; that is, two letters are as far away from each other as the absolute value of the difference between their positions in the alphabet. Then, we have $d(b, a) = d(b, c) = 1$. Now, if we want to transform p into q we simply transform b into a, and the cost is 1. Therefore, we can say that $d(p, r) = 1$, but in order to generate q, we need two (independent) changes and thus $d(p, q) = 1 + 1 = 2$. Under the "classical" quantified bisimulation game, as the maximal disagreement between p and q is also 1, they cannot see that q is more different from p than r is.

Certainly, our "transformational" framework only produces initially (upper) bounds for the distances. instead of a precise value. However, if we consider the whole set of transformations that produce each of the bounds, we can simply define the distance between two processes as the smallest value of those bounds. By the way, it is easy to see that these distances would correspond to the shortest path in the graph defined as follows:

1. Nodes are (abstract) trees (that is, bisimilarity classes) up to commutativity and idempotency.
2. Arcs connect trees that only differ in one arc.
3. The cost of each arc is the distance between the label of the original arc and that at the modified arc.

For the purpose of finding the shortest path in this graph, and hence the distance between trees, we could possibly use the ideas from some of the algorithms in the literature, such as *Dijkstra's Algorithm* [Dij59], *Bellman - Ford's Algorithm* [For56, Moo57, Bel58], or A^* *Search Algorithm* [HNR68]. However, we should be careful because the graph to explore in our case will usually become infinite, and then we would need some clever hints in order to apply the algorithm on the fly to avoid non termination. Nevertheless, we have not explored these algorithmic ways of computing our distances in this thesis.

Our work [RdF12a] started by considering the bisimulation semantics: the canonical representation of processes under this semantics are just (unordered) trees. Then, our distance was based on the idea of how much we needed to modify one of the compared processes (trees) to obtain the other. Once we had our new bisimulation distance, our unified presentation of all the process semantics immediately produced a (unified) notion of distance for each semantics in the *spectrum*. As far as we know, our unified and general definition of the distances corresponding to all the semantics was never proposed before. Certainly, Fahrenberg et al. developed in [FLT11, FL14] a big collection of distances for a copious set of semantics. However, this was done by considering their extensional definitions one by one, thus producing a too colorful set of proposals that became

clearly "debatable" in several of the cases. They simply considered an adequate class of *blind strategies* for the game, but this was somehow done in a too *ad-hoc* way, thus leading them to a quite bit strange (not too accurate in our opinion) set of distances. Instead, the virtues of our definition capacitated us to work in a general way covering all the semantics in the *ltbt-spectrum*, also producing a definition that was consistent with the semantics hierarchy: Whenever we have two different semantics S_1 and S_2 where the latter is finer than the former, we have that the computed distance between any two processes p and q under S_2 will be greater than or equal to the corresponding distance under S_1.

Fig. 3.7 shows our rules to compute the distances as they appear in [RdF12a]. We consider processes up-to bisimulation and the collection of "distance relations" $\{G_m^{\mathcal{L}} \mid m \in \mathbb{N}\}$, where $\mathcal{L}$ is the corresponding semantics under which we define the distance. These relations compute the bounds for the distances, that is: whenever we can generate $pG_n^{\mathcal{L}}q$ we say that the corresponding distance between p and q is not bigger than n.

Definition 10. *Given a semantics $\mathcal{L}$, defined by a preorder $\sqsubseteq_{\mathcal{L}}$, we say that a process q is at global distance at most $m \in \mathbb{N}$ of being better than some other p, w.r.t. the semantics $\mathcal{L}$ and the distance between actions $\overline{d}$, and then we write $gd_{\overline{d}}^{\mathcal{L}}(p,q) \leq n$, if we can infer $p\, G_n^{\mathcal{L}}\, q$, by applying the following rules:*

$$(1)\ \frac{p \sqsubseteq_{\mathcal{L}} q}{p\ G_n^{\mathcal{L}}\ q} \quad (2)\ \frac{p\, G_n^{\mathcal{L}}\, q}{ap\ G_{n+\overline{d}(b,a)}^{\mathcal{L}}\ bq} \quad (3)\ \frac{p\, G_n^{\mathcal{L}}\, p'}{p+q\ G_n^{\mathcal{L}}\ p'+q} \quad (4)\ \frac{p\, G_n^{\mathcal{L}}\, q \quad q\, G_{n'}^{\mathcal{L}}\, r}{p\ G_{n+n'}^{\mathcal{L}}\ r}$$

Figure 3.7: Our distance definition as it was presented in [RdF12a].

Again, an unexpected but quite nice result was obtained when generating our rules: we noticed out a very simple way of expressing the classical bisimulation distance based on the (bi)simulation game [dAMRS08]. This classical distance could be generated by means of the same SOS-rules, just introducing a simple modification in rule (3) –see our paper in Section 6.1–. Furthermore, our work [RdF12a] also presented a coinductive characterization both of bisimulation and simulation game distances. This characterization facilitated us to compute the value not only for finite processes, but for infinite ones, thus obtaining a more general definition. The example presented in Fig. 3.8 gives us an idea of how our rules work in order to generate the (bounds for the) distances between processes. This example considers the processes $p = b + c$ and $q = d + f$. Note in particular that whenever we obtain a certain assert $p'\, G_n^{\mathcal{L}}\, q'$, it will be trivial to obtain any other $p'\, G_m^{\mathcal{L}}\, q'$ with $m > n$ by applying the rule (1), taking $p = q = q'$ and then rule (4).

Example 5. It is easy to check that for the processes in Example 4 and the ready simulation semantics RS, we obtain now the desired distance $gd_d^{RS}(p,q) \leq 4$, since we can infer applying the rules for $\mathcal{L} = RS$ that:

$$\dfrac{\dfrac{\dfrac{b\,G_1^{RS}\,c}{b + c\,G_1^{RS}\,c + c}(3) \quad \dfrac{\dfrac{c + c \sqsubseteq_{RS} c}{c + c\,G_0^{RS}\,c}(1) \quad \dfrac{c\,G_1^{RS}\,d \quad \dfrac{d \sqsubseteq_{RS} d + d}{d\,G_0^{RS}\,d + d}(1)}{c\,G_{1+0}^{RS}\,d + d}(4)}{c + c\,G_{0+1}^{RS}\,d + d}(4)}{b + c\,G_{1+1}^{RS}\,d + d}(4) \quad \dfrac{d\,G_2^{RS}\,f}{d + d\,G_2^{RS}\,f + f}(3)}{b + c\,G_{2+2}^{RS}\,d + f}(4)}{p\ G_4^{RS}\ q}(df)$$

Figure 3.8: An illustrative example presented in [RdF12a].

Based on the *Equational Deduction System*, $\mathbf{DED}(E)$, defined by M. Hennessy in his book *Algebraic Theory of Processes* [Hen88], we developed our paper titled *Distances between Processes: A Pure Algebraic Approach* [RdF12b]. In this paper we presented to the scientific community an algebraic framework to characterize our (bounds for the) distances between processes, in particular covering the case of those semantics that can be axiomatized.

Next, we briefly recall the definitions and the needed terminology in [Hen88] to define $\mathbf{DED}(E)$. A Σ-algebra $\langle A, \Sigma_A \rangle$ satisfies an equation of terms $t \equiv t'$ iff the values of both terms under any valuation are the same. We denote by $\mathcal{C}(E)$, the class of Σ-algebras satisfying the equations E, and then the initial algebra can be presented as a quotient algebra $T_\Sigma / \equiv_E$, for some congruence $\equiv_E$. We can obtain this congruence by means of the equational deduction system $\mathbf{DED}(E)$ presented in Fig. 3.9.

1. Reflexivity $t \equiv t$.
2. Symmetry $t \equiv t' \Rightarrow t' \equiv t$.
3. Transitivity $t \equiv t'$, $t' \equiv t'' \Rightarrow t \equiv t''$.
4. Substitution $t_1 \equiv t'_1, \ldots, t_k \equiv t'_k \Rightarrow f(t_1, \ldots, t_k) \equiv f(t'_1, \ldots, t'_k)$ for every $f \in \Sigma$ of arity k.
5. Instantiation $t \equiv t' \Rightarrow t\rho \equiv t'\rho$ for every substitution ρ.
6. Equations $t \equiv t'$ for every equation $\langle t, t' \rangle \in E$

Figure 3.9: The proof system $\mathbf{DED}(E)$ in [Hen88].

From it, we defined our deduction system for distances $\mathbf{dDED}(E)$ –see Fig. 3.10–, by resetting the clauses that define $\mathbf{DED}(E)$.

Definition 6. *A deduction system for distances, $\mathbf{dDED}(E_{\mathcal{D}})$, between terms in $\mathcal{T}_{\Sigma}(X)$, is a collection of rules including:*

1. *Reflexivity:* $t \equiv_d t$.
2. *Symmetry:* $t \equiv_d t' \Rightarrow t' \equiv_d t$.
3. *Triangular transitivity:* $t \equiv_d t'$, $t' \equiv_{d'} t'' \Rightarrow t \equiv_{d+d'} t''$.
4. *Substitution:* $t_1 \equiv_{d_1} t'_1, \ldots, t_k \equiv_{d_k} t'_k \Rightarrow f(t_1, \ldots, t_k) \equiv_{\widetilde{f}(d_1,\ldots,d_k)} f(t'_1, \ldots, t'_k)$ *for every $f \in \Sigma$ of arity k and the corresponding $\widetilde{f}$ composing distances.*
5. *Instantiation:* $t \equiv_d t' \Rightarrow t\rho \equiv_d t'\rho$, *for every substitution ρ.*
6. *A set of distance equations* $E_{\mathcal{D}} = \{t_i \equiv_{d_i} t'_i \mid i \in I\}$.

Figure 3.10: The deduction systems for distances in [RdF12b].

This definition was intended to be as general as possible. For the case of our distances, the function $\widetilde{f}\ (d_1, \ldots, d_k)$ would be just the sum of the arguments. The set of distance equations will be a characterization of the quotient algebra on top of which we define our distance relations, so that we have $d_i = 0$ in all the cases.

To conclude this section and the work made around our goal 2.1, we present in Fig. 3.11 the formal definition of our algebraic characterizations of the (bounds for the) distances. Note the use of sum in rule 3 (as done in our distance framework), and the fact that we are defining in rule 5, by the system of (in)equations, a preorder and not an equivalence relation; therefore the symmetry is, of course, lost.

The contents of the two papers explained in this section have been thoroughly discussed in several forums during the last year. In April 2014, during my stay in Iceland supported by a *NILS Science and Sustainability scholarship within the project 001-ABEL-CM-2013*, whose responsible is my supervisor, I could present our ideas concerning distances between processes in the *Icelandic Center of Excellence in Theoretical Computer Science (ICE-TCS seminar)* of Reykjavík University, by presenting the seminar [Rom14]. Furthermore, in June 2014, my supervisor shared our ideas in the *Open Problems in Concurrency Theory* held in Bertinoro [dF14], whose aim was to identify the most important open problems in the field of concurrency theory, with the attendance, by invitation, of the most influential researchers in the field of Concurrency Theory.

Definition 8. *Given a semantics $\mathcal{L}$ algebraically defined by means of an axiomatization I on $\leq^{\mathcal{L}}$, and a distance $\overline{d}$ over the set of actions Act, we will say that a process p is at most $d \geq 0$ far away of being better than another process q, w.r.t. the preorder $\leq^{\mathcal{L}}$, if and only if we have $\vdash_{\boldsymbol{dDED}(I)} p \leq^{\mathcal{L}}_{d} q$. $\boldsymbol{dDED}(I)$ is the following deduction system:*

1. $p \leq^{\mathcal{L}}_{d} p$, *for all* $d \in \mathcal{D}$.
2. $p \leq^{\mathcal{L}}_{d_1} p'$ *and* $p' \leq^{\mathcal{L}}_{d_2} p'' \Rightarrow p \leq^{\mathcal{L}}_{d_1+d_2} p''$.
3. *(i)* $p \leq^{\mathcal{L}}_{d_1} p'$ $q \leq^{\mathcal{L}}_{d_2} q' \Rightarrow p + q \leq^{\mathcal{L}}_{d_1+d_2} p' + q'$.
   (ii) $p \leq^{\mathcal{L}}_{d} q \Rightarrow ap \leq^{\mathcal{L}}_{d} aq$, *for all* $a \in Act$.
4. $p \leq^{\mathcal{L}}_{d} p'$ *then* $p\rho \leq^{\mathcal{L}}_{d} p'\rho$, *for every substitution* ρ.
5. $ax \leq^{\mathcal{L}}_{\overline{d}(a,b)} bx$.
   $t \leq^{\mathcal{L}}_{d} t'$, *for every inequation* $t \leq^{\mathcal{L}} t' \in I$.

Figure 3.11: Our definition of the distances as it was presented in [RdF12b].

3.3 Defining distances between infinite processes

The last objective we had overcome (at least in a first approximation) dealt with the development of the distances between infinite processes. Our work [RdF14] was the starting point, and there we defined coinductively our global bisimulation (bounds for the) distance. It can be easily extended to a general coinductive definition for the distances under all the semantics in the *ltbt-spectrum*. In that paper, we called about *trees* when referring to processes because we needed the unfolding of processes into trees in order to manage them. So that, we worked mainly with finitary trees ($FyTrees$), which captured the (bisimulation) semantics of the processes. These trees have finite branches and infinite depth, because of that we needed to consider a coinductive definition of distance.

Now we explain the underlying ideas when applying coinduction to measure the distance between two finitary trees t and t'. Since we are working with trees of infinite depth, if we want to get a bound for the cost of transforming t into t', it would be desirable to have a rule allowing us to change in a single step the "infinite" continuations of a tree (this gave us Rule 3 in Fig. 3.12). Thanks to the "magic of coinduction", whenever we have that two trees $t, t' \in FyTrees$ are at most at distance d, we can assert that $t_1 = at'_1$ are at most at the same distance d from $t_2 = at'_2$; and this can be done without introducing any complex notion of limit in our finite definition. In fact, the bound for the distance turns into αd when the discount factor α is considered.

Following this intuition, together with the basis rules of our distance, that

is, the use of idempotency for free (rule 1 in Fig. 3.12) and the possibility of penalty-changing a single action (rule 2 in Fig. 3.12), in Fig. 3.12 we collect the family of the coinductive transformation between finitary trees.

Definition 8. *Given a domain of actions* $(\mathbb{A}, \mathbf{d})$*, a discount factor* $\alpha \in (0, 1]$ *and a family* $\mathcal{D} \subseteq FyTrees(\mathbb{A}) \times FyTrees(\mathbb{A}) \times \mathbb{R}^+$*, we define the family of relations* $\equiv_d^{\mathcal{D},\alpha}$*, by:*

1. *For all* $d \geq 0$ *we have (i)* $(\sum_{j\in J} a_j t_j) + at + at \equiv_d^{\mathcal{D},\alpha} (\sum_{j\in J} a_j t_j) + at$ *, and (ii)* $(\sum_{j\in J} a_j t_j) + at \equiv_d^{\mathcal{D},\alpha} (\sum_{j\in J} a_j t_j) + at + at$*.*
2. $(\sum_{j\in J} a_j t_j) + at \equiv_{\mathbf{d}(a,b)}^{\mathcal{D},\alpha} (\sum_{j\in J} a_j t_j) + bt$*.*
3. *For all* $(t, t', d) \in \mathcal{D}$ *we have* $(\sum_{j\in J} a_j t_j) + at \equiv_{\alpha d}^{\mathcal{D},\alpha} (\sum_{j\in J} a_j t_j) + at'$*.*

Figure 3.12: The family of coinductive transformations as it presented in [RdF14].

In plain words, we need to use at the first level of the trees the transformations on our definition for finite trees, but instead we can freely use the triples in $\mathcal{D}$ when transforming subtrees at a lower level.

In Fig. 3.13 we show the coinductive proof obligations imposed on the families of triples as above, in order to get satisfactory coinductive families of distances between infinite trees. They were inspired by the conditions imposed on bisimulations, that can be seen as "circular proofs" of bisimilarity for all the pairs in them. In this case we obtain bounds for the distance between the two compared trees by adding all the payments done along the finite coinductive sequence. Thus capturing the full cost of an infinite transformation that remains implicit.

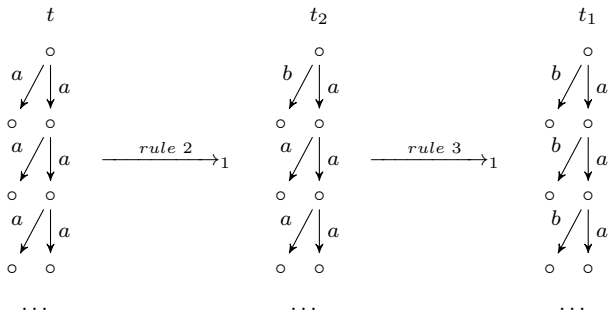
Definition 9. *Given a domain of actions* $(\mathbb{A}, \mathbf{d})$ *and a discount factor* $\alpha \in (0, 1]$*, we say that a family* $\mathcal{D}$ *is an* α*-coinductive collection of distances (*α*-ccd) between finitary trees, if for all* $(t, t', d) \in \mathcal{D}$ *there exists a* finite coinductive transformation sequence $\mathcal{C} := t = t^0 \equiv_{d_1}^{\mathcal{D},\alpha} t^1 \equiv_{d_2}^{\mathcal{D},\alpha} \ldots \equiv_{d_n}^{\mathcal{D},\alpha} t^n = t'$*, with* $d \geq \sum_{j=1}^{n} d_j$*. Then, when there exists an* α*-ccd* $\mathcal{D}$ *with* $(t, t', d) \in \mathcal{D}$*, we will write* $t \equiv_d^{\alpha} t'$*, and say that tree* t *is at most at distance* d *from tree* t' *wrt* α*.*

Notation: We say that the steps generated by application of rules 1 and 2 in Def.8 are *first level steps*; while those generated by rule 3 are *coinductive steps.*

Figure 3.13: Coinductive collections of distances as it presented in [RdF14].

The following example shows how our coinductive definition works. Given $\alpha = 1/2$, a family $\mathcal{D} = \{(t, t_1, 2)\}$, and considering $\mathbf{d}(a, b) = 1$, we can go from t

to t_1 by using the intermediate process t_2, as follows:



If we denote by t' (resp. t'_1, t'_2) the continuations after executing a on the rhs of t (resp. t_1, t_2), then we can apply our rule 2 to prove that $(at') + a\mathbf{0} \equiv^{\mathcal{D},1/2}_{\mathbf{d}(a,b)} (at'_2) + b\mathbf{0}$, getting the bound 1 for the distance between t and t_2. Note that $t'_2 = t$ and $t'_1 = t_1$ and now, since $(t, t_1, 2) \in \mathcal{D}$, we can use rule 3 to prove $(b\mathbf{0}) + at'_2 \equiv^{\mathcal{D},1/2}_1 (b\mathbf{0}) + at'_1$, getting also 1 as a bound for the distance between t_2 and t_1. As a consequence, the finite coinductive sequence $\mathcal{C} := t \equiv^{\mathcal{D},1/2}_1 t_2 \equiv^{\mathcal{D},1/2}_1 t_1$ proves that $\mathcal{D}$ is indeed an 1/2-coinductive collection of distances. Therefore, we have that $t \equiv^{1/2}_2 t_1$, which gives us the bound 2 for the distance between t and t_1.

Our paper [RdF14] also proved the equivalence between our finite framework and the coinductive one, that is, whenever we have two finite trees, the distance between them, computed by using the coinductive approximation, is the same that can be computed using the operational (algebraic) framework.

At this point of our research, we came across what we called in the introduction of this chapter a too exotic place. The trip around the city of infinite trees raised the following question: *Is our coinductive definition of (bounds for the) distance the canonical extension to the previously presented definition for finite processes?* Or equivalently: *Are the (bounds for the) distance obtained by our coinductive definition the uniform bounds for the distances between their finite projections (obtained by cutting trees at a certain finite depth)?* In the beginning, we expected to find a positive answer to this question, and in fact our coinductive definition is always sound wrt the distances between the respective finite approximations. However, when trying to prove the other part of continuity (completeness), stating that any common bound for the distance between (all) the projections of two trees should become a bound for the distance between the given trees, we found, at least at the moment, a dead end. Our multiple essays had indeed nearly proved that desired (and expected) result in several ways. Nevertheless, in all the cases at the

end some annoying technical detail remained impossible to be checked. At the same time, these difficulties kept still far from producing any counterexample.

As good researchers, we did not give up, and we are still immersed in the problem. As commented in the introduction of this chapter, we have been working in collaboration with D. Della Monica trying to find a solution. Unfortunately, we have not get it yet, but we considered that even if unfinished, the results of our efforts should be published in someway. Recently, these partial results were submitted and selected for presentation at the *XV Jornadas de Programación y Lenguajes (PROLE 2015)*, where we presented our paper *Proving Continuity of Coinductive Global Bisimulation Distances: A Never Ending Story* [RdFM15]. We have just received the invitation to submit a revised version, that we expect to be accepted for publication at *Electronic Proceedings in Theoretical Computer Science* in a volume devoted to selected papers presented at *PROLE 2015*.

This paper collected our partial results around the proof of continuity. We tried to prove that whenever we know a common bound for the distances between the respective projections of two trees, then the same bound should also be valid for the distance between those trees. [RdFM15] presented the proof of the lemma stating that the result is true for the two first levels of the trees, by using two different approaches. Unfortunately, for deeper levels it seems necessary an even more complex reasoning that we have not yet discovered. Roughly speaking, our idea was to find a collection of "uniform" operational sequences checking $\mathcal{S}^n := \pi_n(t) \rightsquigarrow_{\alpha,d} \pi_n(t')$, where the operators π_n compute the canonical finite approximations of trees by means of their finite projections. We called these sequences telescopic, and by overlapping all of them, we could get a coinductive sequence proving our desired result for finitary trees, $t \equiv_d^\alpha t'$. Fig. 3.14 presents the definition of those telescopic collection.

Definition 8. *Let $t, t' \in FyTrees(\mathbb{A})$ and let $(\mathscr{S}^n)_{n\in\mathbb{N}}$ be a collection of operational sequences proving $\pi_n(t) \rightsquigarrow_{\alpha,d} \pi_n(t')$. We say that it is* telescopic[2] *if $\pi_m(\mathscr{S}^n) = \mathscr{S}^m$, for all $m \leq n$.*

Any telescopic collection $(\mathscr{S}^n)_{n\in\mathbb{N}}$, produces a "limit" coinductive sequence $\mathscr{C}$ proving $t \equiv_d^\alpha t'$:

Figure 3.14: Definition of telescopic collections as presented in [RdFM15].

The difficulties arose when the given operational sequences $\mathcal{S}^n$ will contain more and more applications of the idempotency equality at the top levels. In such a case they would not constitute a telescopic collection. In particular, it surprised us the fact that sometimes we need to introduce intermediate trees along such a sequence, whose subtrees were wider at some depth than all those corresponding to the source and target trees. Fig. 3.15 presents an example, extracted from our paper [RdFM15], showing this situation.

Example 4. *Let* $\mathbb{A} = \{1,2,3,4,5\}$ *with the "usual" distance* $\mathbf{d}(n,m) = |m-n|$*, for all* $m,n \in \mathbb{A}$*. Let us consider the trees* $t = 1(2+3+4+5)+1(1+2+3+4)$ *and* $t' = 1(1+2+4+5)$*. Then, we have* $t \rightsquigarrow_{1,3} t'$*, which can be obtained by means of the sequence* $\mathscr{S} := t \rightsquigarrow^1_{1,0} 1(2+2+3+4+5)+1(1+2+3+4) \rightsquigarrow^1_{1,1} 1(1+2+3+4+5)+1(1+2+3+4)) \rightsquigarrow^1_{1,0} 1(1+2+3+4+5)+1(1+2+3+4+4) \rightsquigarrow^1_{1,1} 1(1+2+3+4+5)+1(1+2+3+4+5) \rightsquigarrow^1_{1,0} 1(1+2+3+4+5) \rightsquigarrow^1_{1,1} 1(1+2+4+4+5) \rightsquigarrow^1_{1,0} 1(1+2+4+5) = t'$.

Figure 3.15: An important example presented in [RdFM15].

Certainly, if this growth could continue forever (thus becoming arbitrarily large), then we immediately would have a counterexample for our (desired) completeness result. However, we are also far from constructing any such example, and as a matter of fact, we still think that this growth could be somehow controlled. Hence, we could finally obtain the required telescopic sequence and from it the desired result.

As a culmination of this chapter, we present in Fig. 3.16 the timeline of our research, which summarizes our contributions along these years.

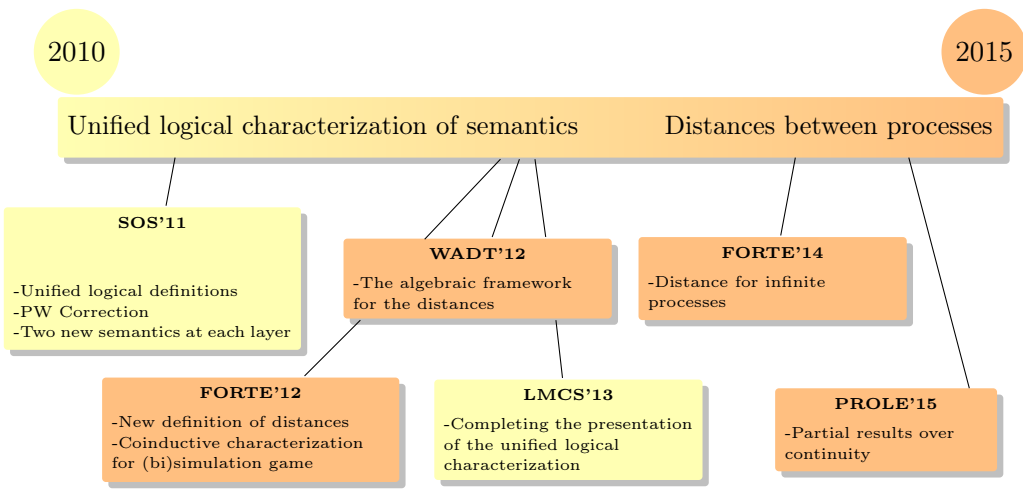


Figure 3.16: The timeline of our research.

chapter 4

Conclusions and future work

"... it would be nice to read it because there's no book so bad that it doesn't have something good in it."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter 59 Part 2

"... será bien leerla, pues no hay libro tan malo que no tenga alguna cosa buena."

The publications summarized in the previous chapter (and presented in the next part of this thesis) are, to our knowledge, a good work in the field of process algebra, and more specifically, in the area related with the equivalence and the distance between processes. This work has not only helped me grow as a researcher, but it also has enabled me to grow as a person. I hope to have established my future within the scientific community, whose first step is actually becoming a reality while writing these concluding lines of my thesis. In the meantime, I also have got my maturity at the personal level and the opportunity of creating a family. Sometimes it had not been easy to conciliate my personal life with the professional challenges, which were quite demanding. However, this period of my life has taught to me that constancy together with the hope of being following the right path, have their reward. A tangible proof is this work, that now has definitely taken shape.

This chapter will conclude with a discussion about the possible ways of continuing our research (confirming somehow my new life as a researcher). First of all, concerning the near future, we present some partial results, intuitions and ideas that we are studying by now, all of them corresponding to the field of distances between processes. At the end, we will present our plan for the future, when we expect to explore new horizons that now are opening in front of us.

4.1 Conclusions

The research that finally have constituted this thesis, started with my wish of better understanding the world of process semantics. Supported by my supervisor, we were looking for methods, notions, and results, that would eventually produce a better comprehension of these semantics. It has been a long trip, but after it we hope to have adequately contributed to the development of processes semantics, and thus to the general development of the processes theory that supports the study of concurrent and reactive systems. These systems study the (possible synchronous) interaction between processes, a concept which is basic in modern computer science, and in general in most of the reality around us.

Due to the way chosen to elaborate this thesis, our results have been already checked and assessed by the corresponding reviewers of the conferences where they were published. Furthermore, our work has started to be known within the scientific community specialized in these topics. A proof of this is that some of our papers have already been cited in several other contributions presented in conferences of our area. We have mentioned before that all of our publications related with the definition of distances [RdF12a, RdF12b, RdF14] are cited in A. Radhakrishnan's thesis [Rad14], but also our work [RdF12a] has been cited by

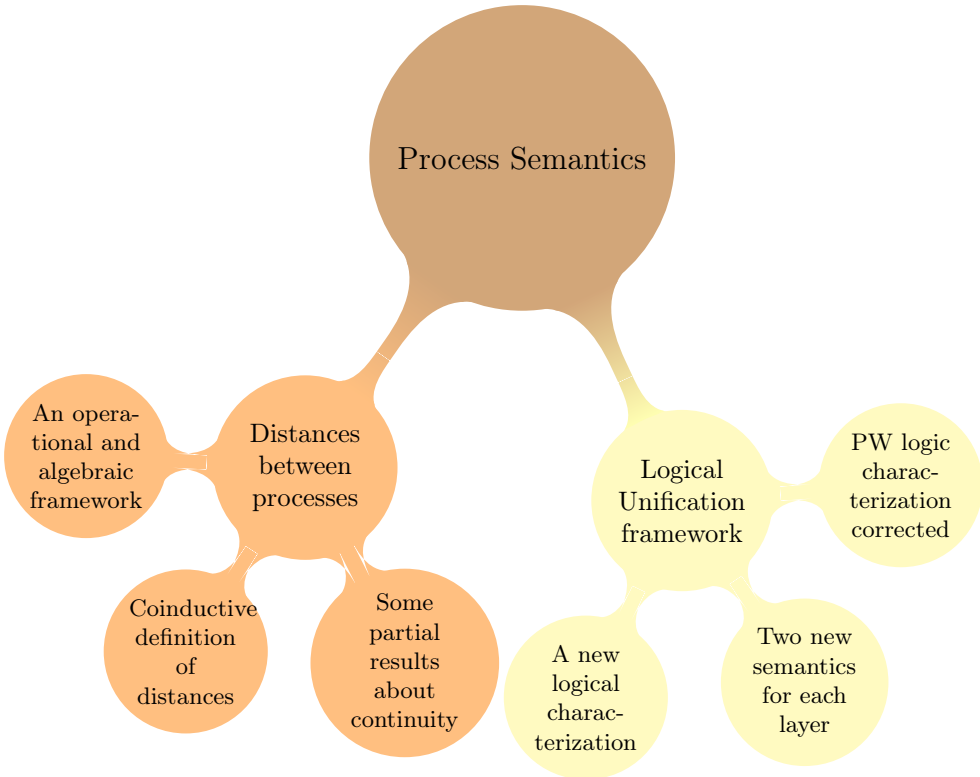


Figure 4.1: The Mind-Map of our results.

C. Gregorio-Rodríguez et al. [GLM13], U. Fahrenberg et al. in [FKLT14], and Giuseppe De Ruvo et al. in [RLM+16]. In this last paper it is also cited our work [RdF14]. Our unification of the *ltbt-spectrum* has been cited by L. Aceto et al. in [AMFI15].

As a summary we show in Fig. 4.1 a nice presentation of our achieved results in the field of process semantics during the last years. All of them have been explained in detail in the previous Chapter 3.

4.2 Future work

As we have commented before, the *NILS project* gave us the opportunity of participating in the interchanges between our university and Reykjavík University. Thank to these stays, we have started some work with Profs. Luca Aceto and

Anna Ingólfsdóttir, and also Ignacio Fábregas, who is now there for a postdoc stay. This work advances the study of distances for processes, taking into account several aspects that play a role in the modeling of processes.

The work developed by Gerald Lüttgen and Walter Vogler titled *Modal Interface Automata* [LV13] is our starting point in the development of a notion of distance between input-output automata. In this work, G. Lüttgen et al. defined a way to translate the information from *Interface Automaton (IA)* to *Disjunctive Modal Transition Systems (dMTS)*, and from that to *Modal Interface Automaton (MIA)*. Roughly speaking, an *IA* is an automaton where the alphabet is partitioned into *Input* and *Output* actions. A *dMTS* is a modal transition system distinguishing between *may* (denoted by $\dashrightarrow$) and *must* (denoted by $\longrightarrow$) transitions which also requires *syntactic consistency*, i.e., $q \xrightarrow{a} Q'$ implies $\forall q' \in Q'$ $q \overset{a}{\dashrightarrow} q'$. Finally, a *MIA* is a *dMTS* with a disjoint alphabet for *Input-Output* actions.

Starting from the refinement relationship for two *dMTS* given in [LV13] and presented below in Def. 16, we are looking for a notion of distance which measures how far away two *dMTS* are from being related by the corresponding equivalence (or preorder) relation.

***Definition* 16 ([LV13])** *Let P, Q be dMTSs. The relation $\mathcal{R} \subseteq P \times Q$ is an* (observational) modal refinement relation *if for all $(p, q) \in \mathcal{R}$:*

- $q \xrightarrow{a} Q'$ *implies* $\exists P'$ *s.t.* $p \xrightarrow{a} P'$ *and* $\forall p' \in P'$ $\exists q' \in Q'$ *with* $(p', q') \in \mathcal{R}$.
- $p \overset{a}{\dashrightarrow} p'$ *implies* $\exists q'$ *s.t.* $q \overset{a}{\dashrightarrow} q'$ *and* $(p', q') \in \mathcal{R}$.

We write $p \sqsubseteq q$ and say that p dMTS-refines *q if there exists an observational modal refinement relation $\mathcal{R}$ such that $(p, q) \in \mathcal{R}$.*

Roughly speaking, a refinement relation on dMTSs allows us to remove may transitions, to turn may transitions into must ones, and to reduce the set of options in a must transition. Our definition should permit these transformations for free, preserving this desirable property: *Two processes related by the refinement relation will be (at most) at distance* 0. Otherwise, our notion of distance should capture the non valid refinement steps, that is: adding a may transition, turning a must transition into a may one, and reaching a bigger disjunctive set in a must transition that are needed to establish the desired relation. Next, we present a sketch of the notion of distance between *dMTS* that we envisage.

***Definition* 17** *Given two dMTSs p and q, a domain of actions $(A \cup \omega, \mathbf{d})$, $m \in \mathbb{R}^+$, and a discount factor $\alpha \in (0, 1]$, we inductively define the* distance steps *on dMTS by:*

1. *For all* $m \geq 0$ *we have* $aP \rightsquigarrow^1_m \mathbf{0}$; $aP \rightsquigarrow^1_m a!P$; $a!P \rightsquigarrow^1_m a!P + ap_i$, *with* $p_i \in P$; $a!P \rightsquigarrow^1_m a!P'$, *with* $P' \subset P$.
2. *For all* $m \geq 0$ $p \rightsquigarrow^1_m p + p$.
3. *If* $p \rightsquigarrow^1_m q$ *then* $p + r \rightsquigarrow^1_m q + r$.
4. *If* $p \rightsquigarrow^1_m q$ *then* $ap \rightsquigarrow^1_{\alpha m} aq$.
5. *For all* $m \geq \mathbf{d}(a, b)$ *we have* $ap \rightsquigarrow^1_m ap + bp$ *and* $a!p \rightsquigarrow^1_m b!p$.
6. *For all* $m \geq \mathbf{d}(\omega, a)$ *we have* $\mathbf{0} \rightsquigarrow^1_m a\mathbf{0}$.
7. *For all* $m \geq \mathbf{d}(a, \omega)$ *we have* $a!P \rightsquigarrow^1_m \mathbf{0}$ *with* $p_i \in P$.
8. *Given* $p_i \in P$ *with* $q \rightsquigarrow^1_m q'$ *then* $a!P \cup \{q\} \rightsquigarrow^1_{\alpha m} a!P \cup \{q'\}$.

We define the family of global distance relations between dMTSs $\langle \rightsquigarrow_d | \; d \in \mathbb{R}^+ \rangle$, *taking* $p \rightsquigarrow_d q$ *if there exists a sequence of distance steps* $\mathcal{S} := p \rightsquigarrow^1_{d_1} p_1 \rightsquigarrow^1_{d_2} \ldots \rightsquigarrow^1_{d_n} p_n = q$ *with* $\sum_{i=1}^n d_i = d$.

Of course, there is still a lot of work to do in order to obtain the definitive notion of distance. As done in [LV13], our notion of distance should empower us by translating this definition to *IA* and *MIA*, while preserving all the good properties. Further references about model interface automata can be found in [RBB+09, RBB+11].

Another research line with L. Aceto, A. Ingólfsdóttir and I. Fábregas concerns the development of a possible definition of distance between the characteristic formulas presented in [SI94, AILS12, AMFI15]. As far as characteristic formulas properly identify each process, we are looking for an adequate definition of distance over formulas in *HML* that could capacitate us to prove the following theorem.

***Theorem* 1** *Given a discount factor* $\alpha \in (0, 1]$, *a (quasi)metric domain* $(\mathbb{A}, \overline{d})$ *and* $p, q \in Proc$, *we have* $d(p, q) = d_1(p, \chi(q))$, *where* $\chi(q)$ *is the characteristic formula of* q, d *is our definition of distances between processes, and* d_1 *is the function that states how far away is a process of satisfying an* HML *formula.*

Again, a sketch for the definition of that function d_1 is presented below:

***Definition* 18** *Given a discount factor* $\alpha \in (0, 1]$ *and the (quasi)metric space* $(\mathbb{A}, \overline{d})$, *we define the distance between a process* p *and a formula* φ, $d_1 : Proc \times HML, \rightarrow \mathbb{D}$ *recursively applying the following rules:*

1. $d_1(p, tt) = 0$ *(bottom element) and* $d_1(p, ff) = \infty$ *(top element).*

2. $d_1(p, \varphi_1 \vee \varphi_2) = min(d_1(p, \varphi_1), d_1(p, \varphi_2))$.

3. $d_1(p, \varphi_1 \wedge \varphi_2) = d_1(p, \varphi_1) + d_1(p, \varphi_2)$.

4. $d_1(p, \langle a \rangle \varphi) = inf\{\overline{d}(\gamma, a) + \alpha d_1(p', \varphi) \mid p \xrightarrow{\gamma} p'\}$ *with* $a \in \mathbb{A}$ *and* γ *a fresh variable in the set of initial actions in* p, $I(p) = \{a \mid p \xrightarrow{a} p'$ *for some* $p'\}$.

5. $d_1(p, [a]\varphi) = \sum_{p \xrightarrow{a} p'} min(\alpha d_1(p', \varphi), inf\{\overline{d}(a, \beta) \mid \beta \in \mathbb{A}\})$ *with* $a \in I(p)$ *and* β *a fresh variable in* $\mathbb{A}$.

As 4. and 5. introduce free variables, we need to solve a finite equational system whose minimal solution will produce the desired distance between p *and* φ.

However, this sketched definition has proved to have some difficulties:

1. It has been introduced in a "too *ad-hoc*" manner, in order to force the desired properties.
2. Certainly, we need to constraint somehow our rule 3 stating that it is only valid whenever φ_1 and φ_2 are independent formulas. Otherwise, for $\varphi_1 = \varphi_2$ we would obtain an impossible equality.

As possible future work, we have in mind to look for notions of distance in the field of testing. Once we have the set of tests that should satisfy a process in order to fulfill an specification, we are wondering *what changes we need to make in a process in order to guarantee that it passes all of these tests.* Another interesting direction could be to define a distance between probabilistic processes following the ideas presented by F. van Breugel et al. in [vBW05] and L. de Alfaro et al. in [dAMRS07, dAMRS08], but using our definition of distance. Of course, we will continue the theoretical study of our coinductive distances, looking for something that could conclude our work on the continuity of our coinductive distance.

There are also some other directions where the introduction of precise measures to compare processes would be useful. For instance, we could compare the quality (and speed) of transmission protocols, something clearly related with the previous work by W. Vogler and G. Lüttgen in [LV06], were faster than preorders where studied. Also, we could continue our research following the work by A. Kiehn and S. Arun-Kumar [KAK05] on amortized bisimulations, possibly taking into account the general approach developed by my supervisor and their collaborators in [dFRG07]. Finally, concerning the logical characterizations of the semantics, we would like to study those for non-interleaving semantics, as those studied in [Gut09, BC14]. We are sure that any of these directions could lead us to new interesting results that will merit to be published.

As a colophon, I would like to conclude with the next quote about research in science, done by Max Nettlau, a historian of the late 19^{th} century:

> *Sometimes malice or stupidity put obstacles to new ideas; hence it is necessary to fight hard to achieve mutual and unconditional tolerance. Only in this way, Science flourishes and progresses, because its foundation is free experimentation and research.*

Bibliography

[Abr89] Samson Abramsky. Observational logic and process semantics (abstract). In *Logic at Botik 1989, Symposium on Logical Foundations of Computer Science, Pereslav-Zalessky, USSR, Proceedings*, volume 363 of *Lecture Notes in Computer Science*, page 1. Springer, 1989.

[Ace03] Luca Aceto. Some of my favourite results in classic process algebra. *Bulletin of the European Association for Theoretical Computer Science*, 81:90–108, 2003.

[AdFGI14] Luca Aceto, David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Anna Ingólfsdóttir. Axiomatizing weak simulation semantics over BCCSP. *Theoretical Computer Science*, 537:42–71, 2014.

[AFI07] Luca Aceto, Wan Fokkink, and Anna Ingólfsdóttir. Ready to preorder: Get your BCCSP axiomatization for free! In *Algebra and Coalgebra in Computer Science, Second International Conference, CALCO 2007, Bergen, Norway, Proceedings*, volume 4624 of *Lecture Notes in Computer Science*, pages 65–79. Springer, 2007.

[AFV01] Luca Aceto, Willem J. Fokkink, and Chris Verhoef. Chapter 3 - structural operational semantics. In *Handbook of Process Algebra*, pages 197 – 292. Elsevier Science, 2001.

[AHK97] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. In *Compositionality: The Significant Difference, International Symposium, COMPOS 1997, Bad Malente, Germany. Revised Lectures*, volume 1536 of *Lecture Notes in Computer Science*, pages 23–60. Springer, 1997.

[AILS12] Luca Aceto, Anna Ingólfsdóttir, Paul Blain Levy, and Joshua Sack. Characteristic formulae for fixed-point semantics: a general framework. *Mathematical Structures in Computer Science*, 22(2):125–173, 2012.

[AMFI15] Luca Aceto, Dario Della Monica, Ignacio Fábregas, and Anna Ingólfsdóttir. When are prime formulae characteristic? In *Mathematical*

Foundations of Computer Science 2015 - 40th International Symposium, MFCS 2015, Milan, Italy, Proceedings, Part I, volume 9234 of *Lecture Notes in Computer Science*, pages 76–88. Springer, 2015.

[Bac54] John W. Backus. The IBM 701 speedcoding system. *Journal of ACM*, 1(1):4–6, 1954.

[Bae05] Jos C. M. Baeten. A brief history of process algebra. *Theoretical Computer Science*, 335(2-3):131–146, 2005.

[BB91] Jos C. M. Baeten and Jan A. Bergstra. Real time process algebra. *Formal Aspects of Computing*, 3(2):142–188, 1991.

[BBK87] Jos C. M. Baeten, Jan A. Bergstra, and Jan Willem Klop. Ready-trace semantics for concrete process algebra with the priority operator. *Computer Journal*, 30(6):498–506, 1987.

[BBR10] Jos C. M. Baeten, Twan Basten, and Michel A. Reniers. *Process Algebra: Equational Theories of Communicating Processes*. Cambridge University Press, New York, NY, USA, 1st edition, 2010.

[BBS95] Jos C. M. Baeten, Jan A. Bergstra, and Scott A. Smolka. Axiomatizing probabilistic processes: ACP with generative probabilities. *Information and Computation*, 121(2):234–255, 1995.

[BC14] Paolo Baldan and Silvia Crafa. A logic for true concurrency. *Journal of ACM*, 61(4):24:1–24:36, 2014.

[BCLvT09] Jos C. M. Baeten, Pieter J. L. Cuijpers, Bas Luttik, and P. J. A. van Tilburg. A process-theoretic look at automata. In *Fundamentals of Software Engineering, Third IPM International Conference, FSEN 2009, Kish Island, Iran, Revised Selected Papers*, volume 5961 of *Lecture Notes in Computer Science*, pages 1–33. Springer, 2009.

[BDP01] Michele Boreale, Rocco De Nicola, and Rosario Pugliese. Divergence in testing and readiness semantics. *Theoretical Computer Science*, 266(1-2):237–248, 2001.

[Bek84] Hans Bekic. The assignment to a type procedure identifier in ALGOL 60. In *Programming Languages and Their Definition - Hans Bekic (1936-1982)*, volume 177 of *Lecture Notes in Computer Science*, pages 2–3. Springer, 1984.

[Bel58] Richard Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16:87–90, 1958.

[Ber01] Jan A. Bergstra. *Handbook of Process Algebra*. Elsevier, New York, NY, USA, 2001.

[BG03] Michele Boreale and Fabio Gadducci. Denotational testing semantics in coinductive form. In *Mathematical Foundations of Computer Science 2003, 28th International Symposium, MFCS 2003, Bratislava, Slovakia, Proceedings*, volume 2747 of *Lecture Notes in Computer Science*, pages 279–289. Springer, 2003.

[BHR84] Stephen D. Brookes, Charles Antony R. Hoare, and Bill Roscoe. A theory of communicating sequential processes. *Journal of ACM*, 31(3):560–599, 1984.

[BIM88] Bard Bloom, Sorin Istrail, and Albert R. Meyer. Bisimulation can't be traced. In *Conference Record of the Fifteenth Annual ACM Symposium on Principles of Programming Languages, San Diego, California, USA*, pages 229–239. ACM Press, 1988.

[BK83] Jan A. Bergstra and Jan Willem Klop. Initial algebra specifications for parametrized data types. *Elektronische Informationsverarbeitung und Kybernetik*, 19(1/2):17–31, 1983.

[BK84] Jan A. Bergstra and Jan Willem Klop. Process algebra for synchronous communication. *Information and Control*, 60(1-3):109–137, 1984.

[BK85] Jan A. Bergstra and Jan Willem Klop. Algebra of communicating processes with abstraction. *Theoretical Computer Science*, 37:77–121, 1985.

[BW90] Jos C. M. Baeten and W. Peter Weijland. *Process Algebra*, volume 18 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, 1990.

[CCH+11] Pavol Cerný, Krishnendu Chatterjee, Thomas A. Henzinger, Arjun Radhakrishna, and Rohit Singh. Quantitative synthesis for concurrent programs. In *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA. Proceedings*, volume 6806 of *Lecture Notes in Computer Science*, pages 243–259. Springer, 2011.

[CCHR14] Pavol Cerný, Martin Chmelik, Thomas A. Henzinger, and Arjun Radhakrishna. Interface simulation distances. *Theoretical Computer Science*, 560:348–363, 2014.

[CD08] Xin Chen and Yuxin Deng. Game characterizations of process equivalences. In *Programming Languages and Systems, 6th Asian Symposium, APLAS 2008, Bangalore, India. Proceedings*, volume 5356 of *Lecture Notes in Computer Science*, pages 107–121. Springer, 2008.

[CE81] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Logics of Programs, Workshop, Yorktown Heights, New York*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer, 1981.

[CHR10] Pavol Cerný, Thomas A. Henzinger, and Arjun Radhakrishna. Quantitative simulation games. In *Time for Verification, Essays in Memory of Amir Pnueli*, volume 6200 of *Lecture Notes in Computer Science*, pages 42–60. Springer, 2010.

[CHR12] Pavol Cerný, Thomas A. Henzinger, and Arjun Radhakrishna. Simulation distances. *Theoretical Computer Science*, 413(1):21–35, 2012.

[CL13] S. Barry Cooper and Jan van Leeuwen. *Alan Turing: His Work and Impact*. Elsevier Science, San Diego, USA, 1st edition, 2013.

[dAFS09] Luca de Alfaro, Marco Faella, and Mariëlle Stoelinga. Linear and branching system metrics. *IEEE Transactions on Software Engineering*, 35(2):258–273, 2009.

[dAHM03] Luca de Alfaro, Thomas A. Henzinger, and Rupak Majumdar. Discounting the future in systems theory. In *Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, Eindhoven, The Netherlands. Proceedings*, volume 2719 of *Lecture Notes in Computer Science*, pages 1022–1037. Springer, 2003.

[dAMRS07] Luca de Alfaro, Rupak Majumdar, Vishwanath Raman, and Mariëlle Stoelinga. Game relations and metrics. In *22nd IEEE Symposium on Logic in Computer Science - LICS 2007, Wroclaw, Poland, Proceedings*, pages 99–108. IEEE Computer Society, 2007.

[dAMRS08] Luca de Alfaro, Rupak Majumdar, Vishwanath Raman, and Mariëlle Stoelinga. Game refinement relations and metrics. *Logical Methods in Computer Science*, 4(3), 2008.

[DBL90] *Theories of Concurrency: Unification and Extension - CONCUR 1990, Amsterdam, The Netherlands, Proceedings*, volume 458 of *Lecture Notes in Computer Science*. Springer, 1990.

[DD09] Yuxin Deng and Wenjie Du. The kantorovich metric in computer science: A brief survey. *Electronic Notes in Theoretical Computer Science*, 253(3):73–82, 2009.

[dF14] David de Frutos-Escrig. On global measures for the evaluation of systems. *Open Problems in Concurrency Theory. Co-sponsored by the* IFIP Working Group 1.8*, Bertinoro, Italy,*, 2014.

[dFG05] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. Bisimulations up-to for the linear time branching time spectrum. In *Concurrency Theory, 16th International Conference - CONCUR 2005, San Francisco, CA, USA, Proceedings*, volume 3653 of *Lecture Notes in Computer Science*, pages 278–292. Springer, 2005.

[dFG08a] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. Constrained simulations, nested simulation semantics and counting bisimulations. *Electronic Notes in Theoretical Computer Science*, 206:41–58, 2008.

[dFG08b] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. Universal coinductive characterisations of process semantics. In *Fifth IFIP International Conference On Theoretical Computer Science - TCS 2008, IFIP 20th World Computer Congress, TC 1, Foundations of Computer Science, Milano, Italy*, volume 273 of *IFIP*, pages 397–412. Springer, 2008.

[dFG09] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. (bi)simulations up-to characterise process semantics. *Information and Computation*, 207(2):146–170, 2009.

[dFGP09a] David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino. On the unification of process semantics: Equational semantics. *Electronic Notes in Theoretical Computer Science*, 249:243–267, 2009.

[dFGP09b] David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino. On the unification of process semantics: Observational semantics. In *Theory and Practice of Computer Science, 35th Conference on Current Trends in Theory and Practice of Computer Science - SOFSEM 2009, Spindleruv Mlýn, Czech Republic. Proceedings*,

volume 5404 of *Lecture Notes in Computer Science*, pages 279–290. Springer, 2009.

[dFGPR13] David de Frutos-Escrig, Carlos Gregorio-Rodríguez, Miguel Palomino, and David Romero-Hernández. Unifying the linear time-branching time spectrum of process semantics. *Logical Methods in Computer Science*, 9(2), 2013.

[dFRG07] David de Frutos-Escrig, Fernando Rosa-Velardo, and Carlos Gregorio-Rodríguez. New bisimulation semantics for distributed systems. In *Formal Techniques for Networked and Distributed Systems - FORTE 2007, 27th IFIP WG 6.1 International Conference, Tallinn, Estonia, Proceedings*, volume 4574 of *Lecture Notes in Computer Science*, pages 143–159. Springer, 2007.

[DGJP99] Josee Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labeled markov systems. In *Concurrency Theory, 10th International Conference - CONCUR 1999, Eindhoven, The Netherlands, Proceedings*, volume 1664 of *Lecture Notes in Computer Science*, pages 258–273. Springer, 1999.

[DGJP04] Josee Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labelled markov processes. *Theoretical Computer Science*, 318(3):323–354, 2004.

[Dij59] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[Dij65] Edsger W. Dijkstra. Solution of a problem in concurrent programming control. *Communication of ACM*, 8(9):569, 1965.

[Dij68a] Edsger W. Dijkstra. Cooperating sequential processes. In *Programming Languages: NATO Advanced Study Institute*. 1968.

[Dij68b] Edsger W. Dijkstra. The structure of "the"-multiprogramming system. *Communication of ACM*, 11(5):341–346, 1968.

[Dij02] Edsger W. Dijkstra. The origin of concurrent programming. In *Cooperating Sequential Processes*, pages 65–138. Springer-Verlag New York, Inc., New York, NY, USA, 2002.

[Dil89] David L. Dill. *Trace theory for automatic hierarchical verification of speed-independent circuits*. ACM distinguished dissertations. MIT Press, 1989.

[DV90] Rocco De Nicola and Frits W. Vaandrager. Action versus state based logics for transition systems. In *Semantics of Systems of Concurrent Processes, LITP Spring School on Theoretical Computer Science, La Roche Posay, France, Proceedings*, volume 469 of *Lecture Notes in Computer Science*, pages 407–419. Springer, 1990.

[Ehr61] Andrzej Ehrenfeucht. An application of games to the completeness problem for formalized theories. *Fundamenta Mathematicae*, 49(2):129–141, 1961.

[EM79] Andrzej Ehrenfeucht and Jan Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8(2):109–113, 1979.

[Fau82] Antony A. Faustini. An operational semantics for pure dataflow. In *Automata, Languages and Programming, 9th Colloquium, Aarhus, Denmark, Proceedings*, volume 140 of *Lecture Notes in Computer Science*, pages 212–224. Springer, 1982.

[FKLT14] Uli Fahrenberg, Jan Kretínský, Axel Legay, and Louis-Marie Traonouez. Compositionality for quantitative specifications. In *Formal Aspects of Component Software - 11th International Symposium, FACS 2014, Bertinoro, Italy, Revised Selected Papers*, volume 8997 of *Lecture Notes in Computer Science*, pages 306–324. Springer, 2014.

[FL13] Uli Fahrenberg and Axel Legay. Generalized quantitative analysis of metric transition systems. In *Programming Languages and Systems - 11th Asian Symposium, APLAS 2013, Melbourne, VIC, Australia. Proceedings*, volume 8301 of *Lecture Notes in Computer Science*, pages 192–208. Springer, 2013.

[FL14] Uli Fahrenberg and Axel Legay. General quantitative specification theories with modal transition systems. *Acta Informatica*, 51(5):261–295, 2014.

[Flo67] Robert W. Floyd. Nondeterministic algorithms. *Journal of ACM*, 14(4):636–644, 1967.

[FLT11] Uli Fahrenberg, Axel Legay, and Claus R. Thrane. The quantitative linear-time–branching-time spectrum. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2011, Mumbai, India*, volume 13 of *LIPIcs*, pages 103–114. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2011.

[For56] Lester R. Ford. Network flow theory. *Santa Monica, California: RAND Corporation*, pages P–923, 1956.

[FPdF07] Ignacio Fábregas, Miguel Palomino, and David de Frutos-Escrig. Reflection and preservation of properties in coalgebraic (bi)simulations. In *Theoretical Aspects of Computing - ICTAC 2007, 4th International Colloquium, Macau, China, Proceedings*, volume 4711 of *Lecture Notes in Computer Science*, pages 231–245. Springer, 2007.

[FTL11] Uli Fahrenberg, Claus R. Thrane, and Kim G. Larsen. Distances for weighted transition systems: Games and properties. In *Proceedings Ninth Workshop on Quantitative Aspects of Programming Languages, QAPL 2011, Saarbrücken, Germany*, volume 57 of *Electronic Proceedings in Theoretical Computer Science*, pages 134–147, 2011.

[GLM13] Carlos Gregorio-Rodríguez, Luis Llana, and Rafael Martínez-Torres. Input-output conformance simulation (iocos) for model based testing. In *Formal Techniques for Distributed Systems - Joint IFIP WG 6.1 International Conference, FMOODS/FORTE 2013, Held as Part of the 8th International Federated Conference on Distributed Computing Techniques, DisCoTec 2013, Florence, Italy. Proceedings*, volume 7892 of *Lecture Notes in Computer Science*, pages 114–129. Springer, 2013.

[GR09] Carlos Gregorio-Rodríguez. *Entendiendo las Semánticas de Procesos.* Phd Thesis. Departamento de Sistemas Informáticos y Computación, UCM, 2009.

[Gue81] Irène Guessarian. *Algebraic Semantics*, volume 99 of *Lecture Notes in Computer Science.* Springer, 1981.

[Gut09] Julián Gutiérrez. Logics and bisimulation games for concurrency, causality and conflict. In *Foundations of Software Science and Computational Structures, 12th International Conference, FOSSACS 2009, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2009*, volume 5504 of *Lecture Notes in Computer Science*, pages 48–62. Springer, 2009.

[Han94] Hans A. Hansson. *Time and Probability in Formal Design of Distributed Systems.* Elsevier Science Inc., New York, NY, USA, 1994.

[Hen88] Matthew Hennessy. Algebraic theory of processes. MIT Press, 1988.

[HF89] Joseph Y. Halpern and Ronald Fagin. Modelling knowledge and action in distributed systems. *Distributed Computing*, 3(4):159–177, 1989.

[Hil96] Jane Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, New York, NY, USA, 1996.

[HJ04] Jesse Hughes and Bart Jacobs. Simulations in coalgebra. *Theoretical Computer Science*, 327(1-2):71–108, 2004.

[HM80] Matthew Hennessy and Robin Milner. On observing nondeterminism and concurrency. In *Automata, Languages and Programming, 7th Colloquium, Noordweijkerhout, The Netherland, Proceedings*, volume 85 of *Lecture Notes in Computer Science*, pages 299–309. Springer, 1980.

[HM85] Matthew Hennessy and Robin Milner. Algebraic laws for nondeterminism and concurrency. *Journal of ACM*, 32(1):137–161, 1985.

[HMS08] Hossein Hojjat, Mohammad Reza Mousavi, and Marjan Sirjani. A framework for performance evaluation and functional verification in stochastic process algebras. In *Proceedings of the 2008 ACM Symposium on Applied Computing - SAC 2008, Fortaleza, Ceara, Brazil*, pages 339–346. ACM, 2008.

[HNR68] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on*, 4(2):100–107, 1968.

[Hoa69] Charles Antony R. Hoare. An axiomatic basis for computer programming. *Communication of ACM*, 12(10):576–580, 1969.

[Hoa78] Charles Antony R. Hoare. Communicating sequential processes. *Communication of ACM*, 21(8):666–677, 1978.

[Hoa85] Charles Antony R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[Jac04] Bart Jacobs. Trace semantics for coalgebras. *Electronic Notes in Theoretical Computer Science*, 106:167–184, 2004.

[Jon89] Bengt Jonsson. A fully abstract trace model for dataflow networks. In *Proceedings of the 16th ACM SIGPLAN-SIGACT Symposium on*

Principles of Programming Languages, POPL 1989, pages 155–165, New York, NY, USA, 1989. ACM.

[Jon94] Bengt Jonsson. A fully abstract trace model for dataflow and asynchronous networks. *Distributed Computing*, 7(4):197–212, 1994.

[KAK05] Astrid Kiehn and S. Arun-Kumar. Amortised bisimulations. In *Formal Techniques for Networked and Distributed Systems - FORTE 2005, 25th IFIP WG 6.1 International Conference, Taipei, Taiwan, Proceedings*, volume 3731 of *Lecture Notes in Computer Science*, pages 320–334. Springer, 2005.

[Kel76] Robert M. Keller. Formal verification of parallel programs. *Communications of the ACM*, 19(7):371–384, 1976.

[Kli04] Bartek Klin. A coalgebraic approach to process equivalence and a coinduction principle for traces. *Electronics Notes in Theoretical Computer Science*, 106:201–218, 2004.

[Kli09] Bartek Klin. Structural operational semantics for weighted transition systems. In *Semantics and Algebraic Specification, Essays Dedicated to Peter D. Mosses on the Occasion of His 60th Birthday*, volume 5700 of *Lecture Notes in Computer Science*, pages 121–139. Springer, 2009.

[Kri63] Saul A. Kripke. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica*, 16(1963):83–94, 1963.

[KS13] Bartek Klin and Vladimiro Sassone. Structural operational semantics for stochastic and weighted transition systems. *Information and Computation*, 227:58–83, 2013.

[LFT11] Kim G. Larsen, Uli Fahrenberg, and Claus R. Thrane. Metrics for weighted transition systems: Axiomatization and complexity. *Theoretical Computer Science*, 412(28):3358–3369, 2011.

[Low95] Gavin Lowe. Probabilistic and prioritized models of timed CSP. *Theoretical Computer Science*, 138(2):315–352, 1995.

[LS89a] Kim G. Larsen and Arne Skou. Bisimulation through probabilistic testing. In *Conference Record of the Sixteenth Annual ACM Symposium on Principles of Programming Languages, Austin, Texas, USA*, pages 344–352. ACM Press, 1989.

[LS89b] Nancy A. Lynch and Eugene W. Stark. A proof of the kahn principle for input/output automata. *Information and Computation*, 82(1):81–92, 1989.

[LS91] Kim G. Larsen and Arne Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991.

[LV06] Gerald Lüttgen and Walter Vogler. Bisimulation on speed: A unified approach. *Theoretical Computer Science*, 360(1-3):209–227, 2006.

[LV13] Gerald Lüttgen and Walter Vogler. Modal interface automata. *Logical Methods in Computer Science*, 9(3), 2013.

[McC60] John McCarthy. Recursive functions of symbolic expressions and their computation by machine, part I. *Communication of ACM*, 3(4):184–195, 1960.

[Mil70a] Robin Milner. Equivalences on program schemes. *Journal of Computer and System Sciences*, 4(3):205–219, 1970.

[Mil70b] Robin Milner. *A Formal Notion of Simulation Between Programs.* University College of Swansea. Department of Computer Science. Computers and Logic Research Group. Memorandum (Swansea). University College of Swansea, 1970.

[Mil71] Robin Milner. An algebraic definition of simulation between programs. In *Proceedings of the 2nd International Joint Conference on Artificial Intelligence. London, UK.*, pages 481–489. William Kaufmann, 1971.

[Mil80] Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science.* Springer, 1980.

[Mil89] Robin Milner. *Communication and concurrency.* PHI Series in computer science. Prentice Hall, 1989.

[Moo57] Edward F. Moore. The shortest path through a maze. *Proceedings of the International Symposium on the Theory of Switching, and Annals of the Computation Laboratory of Harvard University*, pages 285–292, 1957.

[MRG07] Mohammad Reza Mousavi, Michel A. Reniers, and Jan Friso Groote. SOS formats and meta-theory: 20 years after. *Theoretical Computer Science*, 373(3):238–272, 2007.

[MT90] Faron Moller and Chris M. N. Tofts. A temporal calculus of communicating systems. In *Theories of Concurrency: Unification and Extension - CONCUR 1990, Amsterdam, The Netherlands, Proceedings* [DBL90], pages 401–415.

[NC94] Mogens Nielsen and Christian Clausen. Bisimulation, games, and logic. In *Results and Trends in Theoretical Computer Science*, volume 812 of *Lecture Notes in Computer Science*, pages 289–306. Springer, 1994.

[NV09] Sumit Nain and Moshe Y. Vardi. Trace semantics is fully abstract. In *Proceedings of the 24th Annual IEEE Symposium on Logic in Computer Science, LICS 2009, Los Angeles, CA, USA*, pages 59–68. IEEE Computer Society, 2009.

[OdF91] Yolanda Ortega-Mallén and David de Frutos-Escrig. A complete proof system for timed observations. In *Proceedings of the International Joint Conference on Theory and Practice of Software Development - TAPSOFT 1991, Brighton, UK, Volume 1: Colloquium on Trees in Algebra and Programming (CAAP 1991)*, volume 493 of *Lecture Notes in Computer Science*, pages 412–440. Springer, 1991.

[OH86] Ernst-Rüdiger Olderog and Charles Antony R. Hoare. Specification-oriented semantics for communicating processes. *Acta Informatica*, 23(1):9–66, 1986.

[Orl88] James Orlin. A faster strongly polynomial minimum cost flow algorithm. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC 1988, pages 377–387, New York, NY, USA, 1988. ACM.

[Owi76] Susan S. Owicki. A consistent and complete deductive system for the verification of parallel programs. In *Proceedings of the 8th Annual ACM Symposium on Theory of Computing, Hershey, Pennsylvania, USA*, pages 73–86. ACM, 1976.

[Par81] David M. R. Park. Concurrency and automata on infinite sequences. In *Theoretical Computer Science, 5th Gl-Conference*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer, 1981.

[Pet73] Carl Adam Petri. Concepts of net theory. In *Mathematical Foundations of Computer Science: Proceedings of Symposium and Summer*

School, Strbské Pleso, High Tatras, Czechoslovakia, pages 137–146. Mathematical Institute of the Slovak Academy of Sciences, 1973.

[Phi87] Iain Phillips. Refusal testing. *Theoretical Computer Science*, 50(3):241 – 284, 1987.

[Plo76] Gordon D. Plotkin. A powerdomain construction. *SIAM Journal on Computing*, 5(3):452–487, 1976.

[Plo81] Gordon D. Plotkin. A structural approach to operational semantics. *Technical Report DAIMI FN-19, Aarhus University*, 1981.

[Plo04] Gordon D. Plotkin. A structural approach to operational semantics. *The Journal of Logic and Algebraic Programming*, 60-61:17–139, 2004.

[Pnu77] Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA*, pages 46–57. IEEE Computer Society, 1977.

[Rad14] Arjun Radhakrishna. *Quantitative Specifications for Verification and Synthesis*. Phd Thesis. Graduate School of the Institute of Science and Technology Austria, Klosterneuburg, (Austria), 2014.

[RBB+09] Jean-Baptiste Raclet, Eric Badouel, Albert Benveniste, Benoît Caillaud, Axel Legay, and Roberto Passerone. Modal interfaces: unifying interface automata and modal specifications. In *Proceedings of the 9th ACM & IEEE International conference on Embedded software, EMSOFT 2009, Grenoble, France*, pages 87–96. ACM, 2009.

[RBB+11] Jean-Baptiste Raclet, Eric Badouel, Albert Benveniste, Benoît Caillaud, Axel Legay, and Roberto Passerone. A modal interface theory for component-based design. *Fundamenta Informaticae*, 108(1-2):119–149, 2011.

[RdF11] David Romero-Hernández and David de Frutos-Escrig. On the unification of process semantics: Logical semantics. In *Proceedings Eight Workshop on Structural Operational Semantics 2011, SOS 2011, Aachen, Germany*, volume 62 of *Electronic Proceedings in Theoretical Computer Science*, pages 47–61, 2011.

[RdF12a] David Romero-Hernández and David de Frutos-Escrig. Defining distances for all process semantics. In *Formal Techniques for Distributed Systems - Joint 14th IFIP WG 6.1 International Conference, FMOODS 2012 and 32nd IFIP WG 6.1 International Conference, FORTE 2012, Stockholm, Sweden. Proceedings*, volume 7273 of *Lecture Notes in Computer Science*, pages 169–185. Springer, 2012.

[RdF12b] David Romero-Hernández and David de Frutos-Escrig. Distances between processes: A pure algebraic approach. In *Recent Trends in Algebraic Development Techniques, 21st International Workshop, WADT 2012, Salamanca, Spain, Revised Selected Papers*, volume 7841 of *Lecture Notes in Computer Science*, pages 265–282. Springer, 2012.

[RdF14] David Romero-Hernández and David de Frutos-Escrig. Coinductive definition of distances between processes: Beyond bisimulation distances. In *Formal Techniques for Distributed Objects, Components, and Systems - 34th IFIP WG 6.1 International Conference, FORTE 2014, Held as Part of the 9th International Federated Conference on Distributed Computing Techniques, DisCoTec 2014, Berlin, Germany. Proceedings*, volume 8461 of *Lecture Notes in Computer Science*, pages 249–265. Springer, 2014.

[RdFM15] David Romero-Hernández, David de Frutos-Escrig, and Dario Della-Monica. Proving continuity of coinductive global bisimulation distances: A never ending story. In *XV Jornadas sobre Programación y Lenguajes, PROLE 2015 (To appear), Santander, Spain*, Expected to be selected for the issue of EPTCS devoted to the workshop, 2015.

[RLM+16] Giuseppe De Ruvo, Giuseppe Lettieri, Domenico Martino, Antonella Santone, and Gigliola Vaglini. k-bisimulation: a bisimulation for measuring the dissimilarity between processes. In *12^{th} International Conference on Formal Methods of Component Software - FACS 2015 (To appear), Niterói, Brazil*, Lecture Notes in Computer Science. Springer, 2016.

[Rom10] David Romero-Hernández. Caracterizaciones lógicas uniformes de las semánticas de procesos. *Master's thesis, Facultad de Informática, Universidad Complutense de Madrid*, 2010.

[Rom14] David Romero-Hernández. The happiness visiting iceland: Defining distances between processes. *Icelandic Center of Excellence in The-*

oretical Computer Science (ICE-TCS seminar), Reykjavik, Iceland. http://www.ru.is/td/frettir/vidburdir/nr/30747, 2014.

[Ros09] Bill Roscoe. Revivals, stuckness and the hierarchy of CSP models. *The Journal of Logic and Algebraic Programming*, 78(3):163–190, 2009.

[RR88] George M. Reed and Bill Roscoe. A timed model for communicating sequential processes. *Theoretical Computer Science*, 58:249–261, 1988.

[RRS07] Joy N. Reed, Bill Roscoe, and Jane E. Sinclair. Responsiveness and stable revivals. *Formal Aspects of Computing*, 19(3):303–319, 2007.

[RW11] Michel A. Reniers and Tim A. C. Willemse. Folk theorems on the correspondence between state-based and event-based systems. In *Theory and Practice of Computer Science - 37th Conference on Current Trends in Theory and Practice of Computer Science - SOFSEM 2011, Nový Smokovec, Slovakia. Proceedings*, volume 6543 of *Lecture Notes in Computer Science*, pages 494–505. Springer, 2011.

[San09] Davide Sangiorgi. On the origins of bisimulation and coinduction. *ACM Transactions on Programming Languages and Systems*, 31(4), 2009.

[San11] Davide Sangiorgi. *Introduction to Bisimulation and Coinduction*. Cambridge University Press, New York, NY, USA, 2011.

[SI94] Bernhard Steffen and Anna Ingólfsdóttir. Characteristic formulae for processes with divergence. *Information and Computation*, 110(1):149–163, 1994.

[SR11] Davide Sangiorgi and Jan Rutten. *Advanced Topics in Bisimulation and Coinduction*. Cambridge University Press, New York, NY, USA, 1st edition, 2011.

[SS71] Dana Scott and Christopher Strachey. Toward a mathematical semantics for computer languages. Programming Research Group Technical Monograph PRG-6, Oxford University Computing Lab., 1971.

[Sti98] Colin Stirling. The joys of bisimulation. In *Mathematical Foundations of Computer Science 1998, 23rd International Symposium, MFCS*

1998, Brno, Czech Republic, Proceedings, volume 1450 of *Lecture Notes in Computer Science*, pages 142–151. Springer, 1998.

[TFL10] Claus R. Thrane, Uli Fahrenberg, and Kim G. Larsen. Quantitative analysis of weighted transition systems. *The Journal of Logic and Algebraic Programming*, 79(7):689–703, 2010.

[Tho93] Wolfgang Thomas. On the ehrenfeucht-fraïssé game in theoretical computer science. In *Theory and Practice of Software Development, International Joint Conference CAAP/FASE - TAPSOFT 1993, Orsay, France, Proceedings*, volume 668 of *Lecture Notes in Computer Science*, pages 559–568. Springer, 1993.

[Tur36] Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.

[vBSW08] Franck van Breugel, Babita Sharma, and James Worrell. Approximating a behavioural pseudometric without discount for probabilistic systems. *Logical Methods in Computer Science*, 4(2), 2008.

[vBW05] Franck van Breugel and James Worrell. A behavioural pseudometric for probabilistic transition systems. *Theoretical Computer Science*, 331(1):115–142, 2005.

[Ver06] Anatoly M. Vershik. Kantorovich metric: Initial history and little-known applications. *Journal of Mathematical Sciences*, 133(4):1410–1417, 2006.

[vG93] Rob J. van Glabbeek. The linear time - branching time spectrum II. In *4th International Conference on Concurrency Theory - CONCUR 1993, Hildesheim, Germany, Proceedings*, volume 715 of *Lecture Notes in Computer Science*, pages 66–81. Springer, 1993.

[vG01] Rob J. van Glabbeek. The linear time-branching time spectrum I: the semantics of concrete, sequential processes. In *Handbook of Process Algebra, chapter 1*, pages 3–99. Elsevier, 2001.

[Yi90] Wang Yi. Real-time behaviour of asynchronous agents. In *Theories of Concurrency: Unification and Extension - CONCUR 1990, Amsterdam, The Netherlands, Proceedings* [DBL90], pages 502–520.

Part II

Publications

Chapter 5

Logical characterization of processes

"... as it was a very likely and a very feasible thing for him in course of time to come to be an emperor, as he said, or at least an archbishop or some other dignitary of equal rank."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter 26 Part-1

"... que cosa contingente y muy agible era venir con el discurso del tiempo a ser emperador, como él decía, o, por lo menos, arzobispo, u otra dignidad equivalente."

The work presented in this chapter corresponds to the culmination of the classification of the semantics in the *ltbt-spectrum*. The publications collected here constitute the first block of our research.

5.1 The logical characterization in the ltbt-spectrum

In September 2011, the *Eight International Workshop on Structural Operational Semantics (SOS)* was held in Aachen (Germany) co-located with *CONCUR 2011*. There, I presented our investigation about the logical characterization of process semantics. The conference and the Workshop attracted, among others, several respectable researchers such as L. Aceto, W.J. Fokkink, and M. R. Mousavi, that gave me the pleasure of hearing their presentations as well as presenting my work to them.

This paper proposes an alternative logical characterization of the semantics in the *ltbt-spectrum* that is more uniform. It defines the logical constructs that have to be considered in order to logically unify the different semantics.

On the Unification of Process Semantics: Logical Semantics

David Romero-Hernández David de Frutos-Escrig
Facultad CC. Matemáticas, Universidad Complutense de Madrid
Madrid, Spain
Departamento de Sistemas Informáticos y Computación *
dromeroh@pdi.ucm.es defrutos@sip.ucm.es

We continue with the task of obtaining a unifying view of process semantics by considering in this case the logical characterization of the semantics. We start by considering the classic linear time-branching time spectrum developed by R.J. van Glabbeek. He provided a logical characterization of most of the semantics in his spectrum but, without following a unique pattern. In this paper, we present a uniform logical characterization of all the semantics in the enlarged spectrum. The common structure of the formulas that constitute all the corresponding logics gives us a much clearer picture of the spectrum, clarifying the relations between the different semantics, and allows us to develop generic proofs of some general properties of the semantics.

1 Introduction

The definition of the semantics for concurrent / non-deterministic processes is a delicate question. As soon as the effect of non-determinism is taken into account we have to decide to which extent we will do so. Trace semantics, which were adequate for deterministic systems, obviously do not consider non-determinism at all. Instead, bisimulation semantics captures all the information induced by the choices at the observed process. There are different semantics for processes in the literature. The most popular of them were collected in van Glabbeek's linear time-branching time spectrum [6], after being introduced along the years by different authors. At the abstract level a semantics is just an equivalence relation (or a preorder) between processes. These can be defined by choosing between different frameworks for the different semantics, so we have operational, observational, testing, logical and equational semantics.

In [6] we find the famous picture of the ltbt-spectrum (Figure 1) and descriptions of all the semantics in it including observational / testing, logical and equational (when possible) characterizations. Certainly, the basic elements used in the characterizations for a given framework are somewhat related, but a more systematic approach is desirable. In [2, 3], a unified presentation of both the observational and the equational semantics has been developed, and it has been shown how the generic definitions allow to relate both without repeating similar arguments.

In this paper we present a unified view of the logical semantics by showing how different subsets of the Hennessy-Milner logic HML [8] characterize each of the semantics in the spectrum. Certainly, the logical characterizations presented in [6] were also subsets of HML; however in that paper the author looked for sets of formulas as simple (and hence as small) as possible, probably driven by the idea that a smaller set of formulas would make any study based on it simpler. Instead, we will follow the opposite approach. Formally speaking, for each semantics defined by a preorder $\prec$ we have a (larger) language $\mathcal{L} \subseteq HML$ characterizing it, that is defined by $\varphi \in \mathcal{L} \Leftrightarrow ((p \prec q \wedge p \models \varphi) \Rightarrow q \models \varphi)$. However, it is not easy (nor specially illustrative) to look for the whole set of formulas characterizing each of the

*This work was partially supported by the Spanish projects TESIS (TIN2009-14312-C02-01), DESAFIOS10 (TIN2009-14599-C03-01) and PROMETIDOS S2009 / TIC-1465.

M.A. Reniers, P. Sobocinski (Eds.): Workshop on
Structural Operational Semantics 2011 (SOS 2011)
EPTCS 62, 2011, pp. 47–61, doi:10.4204/EPTCS.62.4

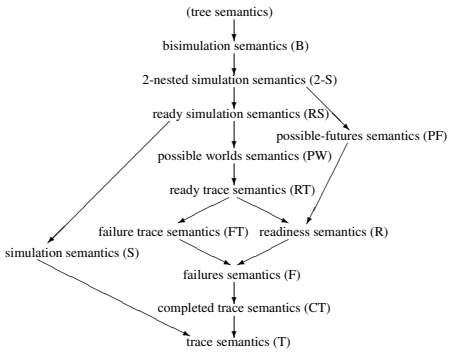


Figure 1: The ltbt-spectrum

semantics: we will consider sufficiently large families defined in a simple way, that provide more natural characterizations which immediately show the relations between the different semantics. For instance, whenever a semantics is finer than other, the logic characterizing the first will contain that for the latter.

As already happened in [2, 3], our unified logical semantics will provide an *enlarged spectrum* (Figure 2) with a clearer structure and additional nodes which correspond to new semantics that in some cases have been also defined using different frameworks by several authors. In particular, we will show the logical characterization of revivals semantics introduced by B. Roscoe in [11], that was already axiomatized in [2].

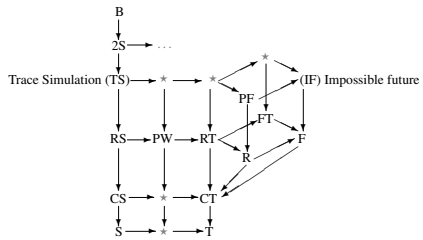


Figure 2: (A part of) the enlarged spectrum

Moreover, we "discover" in this paper the semantics of minimal readies: it was not included in the previous version of the *enlarged spectrum* because the development of the observational and equational frameworks did not detect its existence, while now in the logical framework its definition arises quite naturally. Finally, we have also been able to discover a (minor) mistake in the classic logical characterization of one of the semantics in the original spectrum, Possible Worlds, that has been easily corrected when applying our uniform characterization.

Due to lack of space we had to remove most of the proofs and also a part of the results. An extended version can be found at: `http://maude.sip.ucm.es/~miguelpt/papers/logsem.pdf`.

We strongly appreciate the comments and suggestions of the referees and those from Miguel Palomino, that have contributed to improve the presentation of the paper.

2 Preliminaries

We will not repeat here the long list of original definitions of all the semantics in van Glabbeek's spectrum; please, take a look at [6]. The systematic classification of all these semantics using both observational and equational characterizations can be found at [2, 3]. All the semantics that we consider can be defined over arbitrary (possibly infinite) processes whose operational semantics is defined by means of a labelled transition system (lts) $\mathcal{P} = (Proc, Act, \rightarrow)$. We will use the classical notation $p \xrightarrow{a} p'$ to represent the transitions of processes. Moreover, it is also useful to have a syntactic notation for representing finite processes. We will use BCCSP [6, 2]:

Definition 1 *Given a set of actions Act, the set BCCSP(Act) of processes is defined by the BNF-grammar:* $p ::= \mathbf{0} \mid ap \mid p + q$. *We omit the known operational semantics of BCCSP, which can be found at [6, 2].*

The main ingredient in the classification of semantics, that of course was already present in the original spectrum, is the distinction between branching and linear time semantics. The most important branching semantics are the N-constrained simulations that form the leftmost vertical line of the *enlarged spectrum*. We like to call it the spine of the spectrum, because the rest of the semantics hang on (following the left to right lines) it. N-constrained simulation were studied in a general and systematic way in [4].

Definition 2 *Given a relation N over BCCSP processes, an N-constrained simulation is a relation* S_N *such that* $S_N \subseteq N$ *and whenever* $pS_N q$ *if* $p \xrightarrow{a} p'$ *then there exists* q' *with* $q \xrightarrow{a} q'$ *and* $p' S_N q'$. *We say that p is N-simulated by q, or that q N-simulates p, written* $p \sqsubseteq_{NS} q$, *when there exists an N-constrained simulation* S_N *such that* $pS_N q$.

Although in order to obtain N-constrained similarities with good properties is not necessary for N to be an equivalence relation, that happens in most of the interesting cases (including the most popular ones). For instance, Plain Simulations are just U-constrained simulations, where U is the universal relation $pUq\ \forall p, q \in Proc$. Similarly, Ready Simulations can be defined by means of I-simulations, with $pIq \Leftrightarrow I(p) = I(q) \Leftrightarrow (p \xrightarrow{a} \Leftrightarrow q \xrightarrow{a} \forall\ a \in Act)$; while Complete Simulations correspond to C-simulations, taking $pCq \Leftrightarrow (\exists\ a \in Act\ p \xrightarrow{a} \Leftrightarrow \exists\ a \in Act\ q \xrightarrow{a})$. Note that the Ready Simulation order is usually denoted by $\sqsubseteq_{RS}$, but when using our general notation $\sqsubseteq_{NS}$ we shall write instead $\sqsubseteq_{IS}$.

2.1 Van Glabbeek's logical characterizations for process semantics

Van Glabbeek also presented in [6] a logical characterization of the semantics in the (classical) linear time-branching time spectrum. These logics are sublanguages of the Hennessy-Milner logic [8], $\mathcal{L}_{HM}$, characterizing the bisimulation semantics in the general (possibly infinitary) case.

Definition 3 (Hennessy-Milner logic, HML) *The set* $\mathcal{L}_{HM}$ *of Hennessy-Milner logical formulas is defined by: if* $\varphi, \varphi_i \in \mathcal{L}_{HM}\ \forall i \in I$ *and* $a \in Act$ *then we have* $\bigwedge_{i\in I} \varphi_i$, $a\varphi$, $\neg\varphi \in \mathcal{L}_{HM}$.

For each labelled transition system $\mathbb{P}$, *the satisfaction relation* $\models \subseteq \mathbb{P} \times \mathcal{L}_{HM}$ *is defined by:*

- $p \models a\varphi$ *if there exists* $q \in \mathbb{P} : p \xrightarrow{a} q$ *and* $q \models \varphi$;
- $p \models \bigwedge_{i\in I} \varphi_i$ *if for all* $i \in I : p \models \varphi_i$.
- $p \models \neg\varphi$ *if* $p \not\models \varphi$.

Note that $\bigwedge_{i\in\emptyset} \varphi_i \in \mathcal{L}_{HM}$, and we have $p \models \bigwedge_{i\in\emptyset} \varphi_i$ for all p. Therefore, in the following we will consider that $\top \in \mathcal{L}_{HM}$, where $\top$ is syntactic sugar for $\bigwedge_{i\in\emptyset} \varphi_i$. The finite version of this logic ($\mathcal{L}^f_{HM}$) uses binary conjunction $\wedge$ instead of the general conjunction $\bigwedge_{i\in I}$. It is well known that $\mathcal{L}^f_{HM}$ characterizes the bisimulation semantics between finite image processes, that are those that do not allow infinite branching for any action $a \in Act$ at any state. Van Glabbeek uses $\mathcal{L}_B$ to refer to $\mathcal{L}_{HM}$ in [6].

Formulas \ Semantics ($\mathcal{Z}$)	T	S	CT	CS	F	FT	R	RT	PW	RS	PF	2S	B
$\top \in \mathcal{L}_\mathcal{Z}$	•	ν	•	ν	•	•	•	•	ν	ν	ν	ν	ν
$\mathbf{0} \in \mathcal{L}_\mathcal{Z}$			•	•	ν	ν	ν	ν	ν	ν	ν	ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z}$, $a \in Act \Rightarrow$ $a\varphi \in \mathcal{L}_\mathcal{Z}$	•	•	•	•	•	•	•	•	ν	•	•	•	•
$X \subseteq Act \Rightarrow$ $\widetilde{X} \in \mathcal{L}_\mathcal{Z}$					•	ν	ν	ν	ν	ν	ν	ν	ν
$X \subseteq Act \Rightarrow$ $X \in \mathcal{L}_\mathcal{Z}$							•	ν	•	•	ν	ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z}$, $X \subseteq Act \Rightarrow$ $\widetilde{X}\varphi \in \mathcal{L}_\mathcal{Z}$						•		ν	ν	ν		ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z}$, $X \subseteq Act \Rightarrow$ $X\varphi \in \mathcal{L}_\mathcal{Z}$								•	ν	ν		ν	ν
$\varphi_i \in \mathcal{L}_\mathcal{Z}\ \forall i \in I \Rightarrow$ $\bigwedge_{i\in I}\varphi_i \in \mathcal{L}_\mathcal{Z}$		•		•						•		•	•
$X \subseteq Act$, $\varphi_a \in \mathcal{L}_{PW}\ \forall a \in X \Rightarrow$ $\bigwedge_{a\in X} a\varphi_a \in \mathcal{L}_\mathcal{Z}$									•	ν		ν	ν
$\varphi_i, \varphi_j \in \mathcal{L}_T\ \forall i \in I\ \forall j \in J \Rightarrow$ $\bigwedge_{i\in I}\varphi_i \wedge \bigwedge_{j\in J}\neg\varphi_j \in \mathcal{L}_\mathcal{Z}$											•	ν	ν
$\varphi \in \mathcal{L}_S \Rightarrow$ $\neg\varphi \in \mathcal{L}_\mathcal{Z}$												•	ν
$\varphi \in \mathcal{L}_\mathcal{Z} \Rightarrow$ $\neg\varphi \in \mathcal{L}_\mathcal{Z}$													•

Table 1: Van Glabbeek's logical characterizations for the semantics in the ltbt-spectrum

Definition 4 *Any subset $\mathcal{L}$ of $\mathcal{L}_{HM}$ induces a logical semantics for processes, given by the preorder $\sqsubseteq_\mathcal{L}$: We have $p \sqsubseteq_\mathcal{L} q$ if, and only if, for all $\varphi \in \mathcal{L}$ ($p \models \varphi \Rightarrow q \models \varphi$). We say that $\mathcal{L}$ and $\mathcal{L}'$ are equivalent, and we write $\mathcal{L} \sim \mathcal{L}'$, if they induce the same semantics, that is $\sqsubseteq_\mathcal{L} = \sqsubseteq_{\mathcal{L}'}$.*

Table 1 contains the logical characterization of each of the semantics in van Glabbeek's spectrum: $\mathcal{L}_Z$ with $Z \in \{T, CT, F, FT, R, RT, PF, S, CS, RS, 2S, PW, B\}$, denotes each of the logics; the dots indicate the clauses that we need to introduce to obtain the corresponding languages; and the boxes marked with ν correspond to rules that could be added to $\mathcal{L}_Z$, but they would only introduce redundant formulas. The following connectives, which appear in the table, are not in $\mathcal{L}_{HM}$ but can be obtained as syntactic sugar:

$$\widetilde{X} := \bigwedge_{a\in X} \neg a\top \qquad \widetilde{X}\varphi' := \widetilde{X} \wedge \varphi' \qquad 0 := \widetilde{Act}$$

$$\varphi_1 \wedge \varphi_2 := \bigwedge_{i\in\{1,2\}} \varphi_i \qquad X := \bigwedge_{a\in X} a\top \wedge \bigwedge_{a\notin X} \neg a\top \qquad X\varphi' := X \wedge \varphi' \qquad \widetilde{a} := \neg a\top$$

Disjunction does not appear in $\mathcal{L}_{HM}$, and therefore neither in any of the logics $\mathcal{L}_Z$ characterizing the semantics in the linear time-branching time spectrum. It is probably folklore that it can be added in all cases without changing the expressive power of each of these logics, but since we have not found a clear statement in this direction in any of our references, next we establish the result and comment on its proof.

Proposition 1 *If we define $\mathcal{L}_Z^\vee$ with $Z \in \{T, CT, F, FT, R, RT, PF, S, CS, RS, 2S, PW, B\}$, by adding the clause $\sigma_i \in \mathcal{L}_Z^\vee\ \forall i \in I \Rightarrow \bigvee_{i\in I}\sigma_i \in \mathcal{L}_Z^\vee$ to the clauses which define each semantics $\mathcal{L}_Z$, replacing $\mathcal{L}_Z$ by $\mathcal{L}_Z^\vee$ in each of the other clauses, and making $p \models \bigvee \sigma_i$ iff $\exists i \in I$: $p \models \sigma_i$, then we have $\mathcal{L}_Z^\vee \sim \mathcal{L}_Z$.*

Proof. It is interesting to observe that even if the result is valid for all the semantics, the reason behind is not the same as in the case of bisimulation. In that case, we only need to apply the De Morgan laws to get the "definition" of $\vee$ as a combination of $\neg$ and $\wedge$. However, for the rest of the semantics, we do not have negation as "constructor", but $\vee$ distributes over $\wedge$ and the prefix operator (because $\bigvee a\varphi_i = a \bigvee \varphi_i$),

while negation is never applied to a formula $\varphi' \in \mathcal{L}_Z^\vee$. Therefore, by floating away any $\vee$ in a formula in $\mathcal{L}_Z^\vee$, it becomes equivalent to a disjunction of formulas within the corresponding language $\mathcal{L}_Z$, and then the equivalence of both logics follows.

Remark 1 *Since we have* $\bot = \neg\top = \neg\bigwedge_{i\in\emptyset} = \bigvee_{i\in\emptyset}$ *, we conclude that* $\bot \in \mathcal{L}_Z^\vee$ *, and therefore all the logical semantics defined by these logics remain the same if we add* $\bot$ *and disjunction to their definitions. Moreover,* $\wedge$ *cannot be filtered by the prefix operator. By the way, this makes the difference between linear semantics (whose logics do not allow an arbitrary use of conjunction) and branching semantics (where we can arbitrarily use conjunction). It is important to note that* $a\bot \sim \bot$ *and therefore* $a\bot \nsim \neg a\top$.

2.2 Observational characterizations for process semantics

There is a clear connection between the observational and the logical semantics. In fact, we expected that once we had a unified presentation of the observational semantics it would be easy to transmute it into a unified presentation of the logical semantics. This was not that easy at the end, but certainly our unified logics were inspired by the previously obtained unified observational semantics. Moreover, we need these definitions if we want to check that our new logical semantics are indeed equivalent characterizations of the same semantics. Obviously, for the cases of the semantics in the classic spectrum we could instead compare (one by one) our new logics and those provided by van Glabbeek in [6], but this cannot be done for any of the new semantics. Therefore, we briefly present next the definitions (from [3]) needed to get these observational characterizations.

One important fact about these characterizations is its finite character. All the considered observations are (structurally) finite, and this means that the characterizations work as long as we keep ourselves to the continuous side of the range of possible semantic domains. Therefore, we have to restrict ourselves to finite processes, or at least to image-finite processes. It is for this class of processes that Th. 1 works.

Definition 5 *The sets* L_N *of local observations corresponding to each of the N-constrained simulations in the spectrum, and* $L_N(p)$ *of observations associated to a process p, are defined as follows:*

- *S:* $L_U = \{\cdot\}$, $L_U(p) = \cdot$.
- *CS:* $L_C = Bool$, $L_C(p)$ *is true if* $p \models \mathbf{0}$ *and false otherwise.*
- *RS:* $L_I = \mathcal{P}(Act)$, $L_I(p) = I(p) = \{a |\ a \in Act \text{ and } p \xrightarrow{a}\}$.
- *TS:* $L_T = \mathcal{P}(Act^*)$, $L_T(p)$ *is T(p), the set of traces of p.*
- *2S:* $L_S = \{\|p\|_S\}$, $L_S(p) = \|p\|_S$ *where* $\|p\|_S$ *denotes the simulation equivalence class of p.*
- *kS:* $L_S = \{\|p\|_{(k-1)S}\}$, $L_S(p) = \|p\|_{(k-1)S}$, *where* $\|p\|_{kS}$ *denotes the k-nested simulation equivalence class of p.*

Each $N \in \{U, C, I, T, S\}$ induces uniformily an equivalence relation, that by abuse of notation we will also denote by N: $pNq ::= L_N(p) = L_N(q)$.

Remark 2 *In the definition above we have considered both the trace semantics and the simulation semantics when defining* L_T *and* L_S*. Certainly, we expect that the reader will be familiarized with these two classic semantics, and this is why we avoid a reminder of their definitions here. Also, there is another (more formal) reason for which we do this: the trace and the simulation semantics are two of the semantics to be classified by our systematic approaches, and it would not be nice to have their definitions in advance. Instead, we can apply (when needed) our definitions in a sliced way: based on U we define plain simulations, and then the trace semantics, and once this is done, we have T and S to define TS and 2S. The same is valid, step by step, for all the nested simulation semantics.*

Definition 6 *1. A branching general observation (bgo for short) of a process is a finite, non-empty tree whose arcs are labeled with actions in Act and whose nodes are labeled with local observations from L_N, for N a constraint; the corresponding set BGO_N is recursively defined as: $\langle l, \emptyset\rangle \in BGO_N$ for $l \in L_N$; $\langle l, \{(a_i, bgo_i) \mid i \in 1..n\}\rangle \in BGO_N$ for every $n \in \mathbb{N}, a_i \in Act$ and $bgo_i \in BGO_N$.*

2. The set $BGO_N(p)$ of bgo's of a process p corresponding to the constraint N is $BGO_N(p) = \{\langle L_N(p), S\rangle \mid S \subseteq \{(a, bgo) | bgo \in BGO_N(p'), p \xrightarrow{a} p'\}\}$. We write $p \leq^b_N q$ if $BGO_N(p) \subseteq BGO_N(q)$.

Theorem 1 ([3]) *For all $N \in \{U, C, I, T, S\}$ and any two processes p and q, $p \sqsubseteq_{NS} q$ iff $p \leq^b_N q$.*

Definition 7 *1. The set LGO_N of linear general observations (lgo for short) for the set of local observations L_N is the subset of BGO_N defined as: $\langle l, \emptyset\rangle \in LGO_N$ for each $l \in L_N$; $\langle l, \{(a, lgo)\}\rangle$ whenever $a \in Act$ and $lgo \in LGO_N$.*

2. The set $LGO_N(p)$ of lgo's of a process p with respect to the set of local observations L_N is $LGO_N(p) = BGO_N(p) \cap LGO_N$.

Definition 8 *For $\zeta, \zeta' \subseteq LGO_N$, we define the orders $\leq^l_N$, $\leq^{l\supseteq}_N$, $\leq^{lf}_N$, and $\leq^{lf\supseteq}_N$ by:*

- $\zeta \leq^l_N \zeta' \stackrel{def}{\Leftrightarrow} \zeta \subseteq \zeta'$.
- $\zeta \leq^{l\supseteq}_N \zeta' \stackrel{def}{\Leftrightarrow} \forall\, X_0 a_1 X_1 \ldots X_n \in \zeta\ \exists\, Y_0 a_1 Y_1 \ldots Y_n \in \zeta'\ \forall i \in 0..n\ X_i \supseteq Y_i$.
- $\zeta \leq^{lf}_N \zeta' \stackrel{def}{\Leftrightarrow} \forall\, X_0 a_1 X_1 \ldots X_n \in \zeta\ \exists\, Y_0 a_1 Y_1 \ldots Y_n \in \zeta'\ X_n = Y_n$.
- $\zeta \leq^{lf\supseteq}_N \zeta' \stackrel{def}{\Leftrightarrow} \forall\, X_0 a_1 X_1 \ldots X_n \in \zeta\ \exists\, Y_0 a_1 Y_1 \ldots Y_n \in \zeta'\ X_n \supseteq Y_n$.

Definition 9 *Given two processes p and q and $Z \in \{l, l\supseteq, lf, lf\supseteq\}$, we write $p \leq^Z_N q$ iff $LGO_N(p) \leq^Z_N LGO_N(q)$. We will denote the corresponding equivalence by $=^Z_N$.*

In the cases in which there is no previously known (equivalent) definition for our new semantics, the definition above will give us "the" definition of each one of these new semantics; instead, each of the linear semantics in the old spectrum has a companion in our *enlarged spectrum*. For instance, the linear semantics in the diamond to the right of RS (see Figure 2) satisfy the following theorem.

Theorem 2 *(1) $p \sqsubseteq_{RT} q$ iff $p \leq^l_I q$; (2) $p \sqsubseteq_{FT} q$ iff $p \leq^{l\supseteq}_I q$; (3) $p \sqsubseteq_R q$ iff $p \leq^{lf}_I q$; (4) $p \sqsubseteq_F q$ iff $p \leq^{lf\supseteq}_I q$.*

3 A new logical characterization of the most popular semantics

Next we will present in a uniform way the new logics that characterize the different semantics. Each of them is defined by a set of rules, and as usual we assume that only the formulas that can be obtained by finite application of these rules are in the defined logics. We begin by studying the particular cases of the best known classical semantics, that is, those at the layer of Ready Simulation in the *enlarged spectrum*. All of them use in some way the set of formulas $\mathcal{L}_I = \{a\top \mid a \in Act\}$ that characterizes the initial offers of a process. In Section 4, we will present the logics for the rest of the semantics in a unified way.

Definition 10 ***Ready Simulation semantics (RS)**: we define the set of formulas $\mathcal{L}'_{RS}$ for ready simulation semantics by $\sigma \in \mathcal{L}_I \Rightarrow \sigma \in \mathcal{L}'_{RS}$; $\sigma \in \mathcal{L}_I \Rightarrow \neg\sigma \in \mathcal{L}'_{RS}$; $\varphi_i \in \mathcal{L}'_{RS}\, \forall i \in I \Rightarrow \bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{RS}$; $\varphi \in \mathcal{L}'_{RS}$, $a \in$ Act $\Rightarrow a\varphi \in \mathcal{L}'_{RS}$;.*

***Ready traces semantics (RT)**: we define the set of formulas $\mathcal{L}'_{RT}$ for ready trace semantics by $\top \in \mathcal{L}'_{RT}$; $\varphi \in \mathcal{L}'_{RT}, X_1, X_2 \subseteq \mathcal{L}_I \Rightarrow (\bigwedge_{a\top\in X_1} a\top \wedge \bigwedge_{b\top\in X_2} \neg b\top) \wedge \varphi \in \mathcal{L}'_{RT}$; $\varphi \in \mathcal{L}'_{RT}$, $a \in$ Act $\Rightarrow a\varphi \in \mathcal{L}'_{RT}$.*

***Failure traces semantics (FT)**: we define the set of formulas $\mathcal{L}'_{FT}$ for failure traces semantics by $\top \in \mathcal{L}'_{FT}$; $\varphi \in \mathcal{L}'_{FT}, X_1 \subseteq \mathcal{L}_I \Rightarrow (\bigwedge_{a\top\in X_1} \neg a\top) \wedge \varphi \in \mathcal{L}'_{FT}$; $\varphi \in \mathcal{L}'_{FT}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{FT}$.*
***Readiness semantics (R)**: we define the set of formulas $\mathcal{L}'_R$ for readiness semantics by $\top \in \mathcal{L}'_R$; $X_1, X_2 \subseteq \mathcal{L}_I \Rightarrow (\bigwedge_{a\top\in X_1} a\top \wedge \bigwedge_{b\top\in X_2} \neg b\top) \in \mathcal{L}'_R$; $\varphi \in \mathcal{L}'_R$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_R$.*
***Failures semantics (F)**: we define the set of formulas $\mathcal{L}'_F$ for failures semantics by $\top \in \mathcal{L}'_F$; $X_1 \subseteq \mathcal{L}_I \Rightarrow (\bigwedge_{a\top\in X_1} \neg a\top) \in \mathcal{L}'_F$; $\varphi \in \mathcal{L}'_F$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_F$.*

One can immediately check in the definition above that $\mathcal{L}'_{RS} \subseteq \mathcal{L}_B$, thus obtaining that Ready Simulation semantics is coarser than Bisimulation equivalence. We also have $\mathcal{L}'_F \subseteq \mathcal{L}'_R$, $\mathcal{L}'_F \subseteq \mathcal{L}'_{FT}$, $\mathcal{L}'_R \subseteq \mathcal{L}'_{RT}$, $\mathcal{L}'_{FT} \subseteq \mathcal{L}'_{RT}$ and $\mathcal{L}'_{RT} \subseteq \mathcal{L}'_{RS}$, which can be interpreted in the same way. Let us now focus our attention on the third rule of the definition of $\mathcal{L}'_{RS}$: the unrestricted use of conjunction corresponds to the branched character of the semantics. Moreover, the two first rules allow us to fix the set of offers at the states of the process as I-simulations impose. Instead, the linear semantics only allow the use of conjunction to join the simple formulas that permit us to fix the set of offers along a computation in the case of the readies-based semantics, or their over-approximations (obtained by means of the negated formulas $\neg a\top$), in the case of the failures-based semantics. Finally, notice how these simple formulas can only be checked at the end, for the simpler coarser semantics.

Now, for $X \in \{RS, RT, FT, R, F\}$ we can prove that each of the logics, $\mathcal{L}'_X$, is a superset of the corresponding logic, $\mathcal{L}_X$, defined by van Glabbeek in [6]. To be precise, for the cases of FT and F semantics we need to remove the syntactic sugar used by van Glabbeek.

Proposition 2 *1. $\mathcal{L}'_{RS} \supseteq \mathcal{L}_{RS}$. We also have $\mathcal{L}_{RS} \subsetneq \mathcal{L}'_{RS}$.*

2. *$\mathcal{L}'_{RT} \supseteq \mathcal{L}_{RT}$. We also have $\mathcal{L}_{RT} \subsetneq \mathcal{L}'_{RT}$.*
3. *$\mathcal{L}'_{FT} \supseteq$ desugared($\mathcal{L}_{FT}$), where the desugaring function removes the syntactic sugar used in $\mathcal{L}_{FT}$.*
4. *$\mathcal{L}'_R \supseteq \mathcal{L}_R$. We also have $\mathcal{L}_R \subsetneq \mathcal{L}'_R$.*
5. *$\mathcal{L}'_F \supseteq$ desugared($\mathcal{L}_F$), where the desugared function removes the syntactic sugar used in $\mathcal{L}_F$.*

Proof. All of them are simple and similar, so we will only present the proof of 2.

- 2| To prove that $\mathcal{L}'_{RT} \supseteq \mathcal{L}_{RT}$ it is sufficient to show that for every $X \subseteq Act$ and any $\varphi \in \mathcal{L}_{RT}$, the formula $(\bigwedge_{af\in X} a\top \wedge \bigwedge_{b\notin X} \neg b\top) \wedge \varphi$ belongs to $\mathcal{L}'_{RT}$. Note that $b \notin X$ is equivalent to $b \in \overline{X}$, so taking $X_1 = X$ and $X_2 = \overline{X}$ we have that the considered formula belongs to $\mathcal{L}'_{RT}$. To prove that $\mathcal{L}_{RT} \subset \mathcal{L}'_{RT}$, it is sufficient to note that $(\neg b\top) \wedge \varphi$ belongs to $\mathcal{L}'_{RT}$, by simply taking $X_1 = \emptyset$ and $X_2 = \{b\}$, but it does not belong to $\mathcal{L}_{RS}$.

We have said in our Introduction that our logics are chosen as large as necessary, to obtain more natural characterizations. This is why, in most of the cases, we have obtained a logic larger than that proposed by van Glabbeek. In order to prove the equivalences between ours and van Glabbeek's logics, we have to show that the new formulas that we included in our logics are in fact redundant.

Proposition 3 *We have (1) $\mathcal{L}_{RS} \sim \mathcal{L}'_{RS}$; (2) $\mathcal{L}_{RT} \sim \mathcal{L}'_{RT}$; (3) $\mathcal{L}_{FT} \sim \mathcal{L}'_{FT}$; (4) $\mathcal{L}_R \sim \mathcal{L}'_R$ and (5) $\mathcal{L}_F \sim \mathcal{L}'_F$.*

Proof. As above we will only present one of the proofs.

- 2| We have seen that the formulas in $\mathcal{L}_{RT}$ are particular cases of the formulas in $\mathcal{L}'_{RT}$, those that totally define the offers at the states along a computation (when we apply the second clause in the definition of $\mathcal{L}'_{RT}$ taking $X_2 = \overline{X_1}$). Instead, our more general formulas $(\bigwedge_{a\top\in X_1} a\top \wedge \bigwedge_{b\top\in X_2} \neg b\top) \wedge$

φ, where $\varphi \in \mathcal{L}'_{RT}$, could give us some partial information, combining both positive information $a\top \in X_1$ and negative information $b\top \in X_2$, which tells us that we are in an arbitrary state X, satisfying $X_1 \subseteq X \subseteq \overline{X_2}$. But we can replace these formulas by the disjunction of all the formulas describing any of these possible offers X. By repeating this procedure at each level of the formula, we finally obtain a disjunction of formulas in $\mathcal{L}_{RT}$. To conclude, it is enough to apply Prop. 1.

In the following, when we consider a logic $\mathcal{L}_Z$ and the index Z refers to some concrete semantics, as is the case with RS, RT, FT, R, F above, by abuse of notation we will simply write $\sqsubseteq'_Z$ instead of $\sqsubseteq_{\mathcal{L}'_Z}$ when referring to the preorder induced by the logic $\mathcal{L}'_Z$.

Theorem 3 *1. The logical semantics $\sqsubseteq'_{RS}$ induced by the logic $\mathcal{L}'_{RS}$ is equivalent to the observational branching semantics defined by $\leq^b_I$, generated by the set of branching general observations BGO_I.*

2. The logical semantics $\sqsubseteq'_{RT}$ (resp. $\sqsubseteq'_{FT}$, $\sqsubseteq'_R$, $\sqsubseteq'_F$) induced by the logic $\mathcal{L}'_{RT}$ (resp. $\mathcal{L}'_{FT}$, $\mathcal{L}'_R$, $\mathcal{L}'_F$) is equivalent to the observational linear semantics defined by the domain of linear general observations LGO_I, ordered by $\leq^l_I$ (resp. $\leq^{l\supseteq}_I$, $\leq^{lf}_I$, $\leq^{lf\supseteq}_I$,) defined at Def. 8.

Proof. It is a consequence of Prop. 3, the results by van Glabbeek collected in Table 1, Th. 1 and Th. 2.

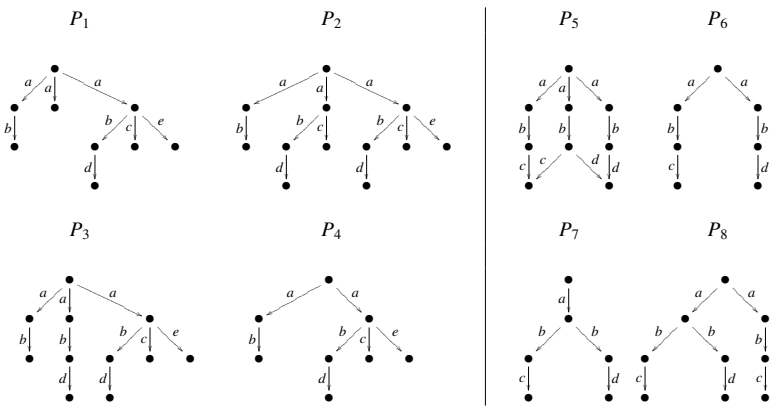


Figure 3: Example to show the strength of the different logics

Example 1 *Figure 3 shows a collection of examples to illustrate the differences between the semantics in the layer of RS at the spectrum. All the stated equivalences can be checked by taking any arbitrary formula from the logic defining each of the semantics. For readability, we omit the last $\top$ in all subformulas. Besides, $\sim_X$, (resp. $\nsim_X$) , where X is a set of indexes, represents any $\sim_Z$ (resp. $\nsim_Z$), with Z $\in$ X.*

- $P_1 \not\sqsubseteq'_F P_2$, *and then* $P_1 \not\sqsubseteq'_{\{R,\,FT,\,RT,\,RS\}} P_2$*; this is because* $P_1 \models a(\neg b \wedge \neg c)$, *but* P_2 *does not.*
- $P_2 \sim_F P_3$, *but* $P_2 \not\sqsubseteq'_{\{R,\,FT\}} P_3$ *and then* $P_2 \not\sqsubseteq'_{\{RT,\,RS\}} P_3$, *using that* $P_2 \models a(\neg e \wedge c)$, *but* P_3 *does not.*
- $P_3 \sim_{\{F,\,R\}} P_4$, *but* $P_3 \not\sqsubseteq'_{FT} P_4$ *and then* $P_3 \not\sqsubseteq'_{\{RT,\,RS\}} P_4$, *because* $P_3 \models a(\neg c \wedge b(\neg e \wedge d))$, *but* P_4 *does not.*
- $P_5 \sim_{\{F,\,FT\}} P_6$, *but* $P_5 \not\sqsubseteq'_R P_6$ *and then* $P_5 \not\sqsubseteq'_{\{RT,\,RS\}} P_6$, *using that* $P_5 \models ab(c \wedge d)$, *but* P_6 *does not.*
- $P_6 \sim_{\{F,\,R,\,RT,\,FT\}} P_7$, *but* $P_7 \not\sqsubseteq'_{RS} P_6$, *using that* $P_7 \models a(bc \wedge bd)$, *but* P_6 *does not.*
- $P_7 \sim_{\{F,\,R,\,RT,\,FT,\,RS\}} P_8$.

4 Our new unified logical characterizations of the semantics

Inspired by the semantics studied in Section 3, next we define the general format for the logics characterizing each of the semantics in the *enlarged spectrum*. We start by enlarging the spectrum a bit more, to include all the elements needed to characterize the rest of the semantics in a systematic way.

Definition 11 1. ***Universal semantics (U)**: We define the set of Universal formulas, $\mathcal{L}'_U$, that characterizes the trivial semantics that identifies all the processes, by $\mathcal{L}'_U = \{\top\}$.*

2. ***Complete semantics (C)**: It is defined by $\sqsubseteq_C$, taking $p \sqsubseteq_C q ::= (p \xrightarrow{a} \Rightarrow \exists\, b \in Act\ q \xrightarrow{b})$. That is, it only distinguishes terminated processes (equivalent to $\mathbf{0}$) from non-terminated ones. We define the set of Complete formulas $\mathcal{L}'_C$ characterizing it, by $\mathcal{L}'_C = \{\top, \neg 0\}$.*

3. ***Initial offer semantics (I)**: It is defined by $\sqsubseteq_I$, taking $p \sqsubseteq_I q ::= I(p) \subseteq I(q)$. That is, it only observes the set of initial actions of a process, $I(p) = \{a \mid a \in Act \wedge p \xrightarrow{a}\}$. We define the set of Initial offer formulas $\mathcal{L}'_I$ characterizing it, by $\mathcal{L}'_I = \{\top, \neg 0\} \cup \{a\top \mid a \in \mathrm{Act}\}$.*

In the definition above the sub-formula $\neg 0$ is just syntactic sugar for the formula $\neg(\bigwedge_{a\in Act} \neg a\top)$. Therefore, all these new logics are indeed sublogics of $\mathcal{L}_{HM}$, and we do not need to define their semantics.

Note that $\mathcal{L}'_I$ is a bit larger than the logic $\mathcal{L}_I$ used in Section 3. Once again, this is so in order to get a more uniform presentation of our logics: $\neg 0$ is indeed redundant. As a consequence, we immediately obtain that the Complete semantics is coarser than the Initial offer semantics, because $\mathcal{L}'_C \subseteq \mathcal{L}'_I$. Based on this result we will also easily obtain that the Complete Simulation is coarser than the Ready Simulation.

4.1 The simulation semantics

As discussed in [3], the simulation semantics constitute the spine of the new spectrum. Moreover, all of them are defined in a homogeneous way using the notion of constrained simulation from [4].

Definition 12 *Given a set of formulas $\mathcal{L}'_N$ defining a semantics N, we define the set of formulas $\mathcal{L}'_{NS}$ that defines the N-constrained simulation semantics by $\sigma \in \mathcal{L}'_N \Rightarrow \sigma \in \mathcal{L}'_{NS}$; $\sigma \in \mathcal{L}'_N \Rightarrow \neg\sigma \in \mathcal{L}'_{NS}$; $\varphi_i \in \mathcal{L}'_{NS} \forall i \in I \Rightarrow \bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{NS}$; $\varphi \in \mathcal{L}'_{NS}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{NS}$.*

Taking $N \in \{U, C, I\}$ we obtain $\mathcal{L}'_{US}$, $\mathcal{L}'_{CS}$ and $\mathcal{L}'_{IS}$, that in the first and last cases we rewrite as $\mathcal{L}'_S$ and $\mathcal{L}'_{RS}$, respectively, in order to emphasize the classic notation for simulation semantics. Once that we have $\mathcal{L}'_S$ we can also obtain $\mathcal{L}'_{SS}$, that we will also denote as $\mathcal{L}'_{2S}$. To complete the collection of simulation semantics we will only need $\mathcal{L}'_{TS}$, that will be based on $\mathcal{L}'_T$, to be defined in the next section.

If we compare the definition above with the particular case of Ready Simulation in Def. 10, the differences concern the two first rules, by means of which we impose that the process will traverse states which are in the corresponding N-equivalence class all along the tree of computations checked by a formula in $\mathcal{L}'_{NS}$. Next we state the equivalence between our logics and those by van Glabbeek in [6].

Proposition 4 *We have (1) $\mathcal{L}'_S \sim \mathcal{L}_S$, (2) $\mathcal{L}'_{CS} \sim \mathcal{L}_{CS}$ and (3) $\mathcal{L}'_{2S} \sim \mathcal{L}_{2S}$.*

4.2 Logical characterization of the linear semantics

We start by defining the closure operators, by means of which we are able to express to which extent conjunction and negation can be used at the logical characterizations of each of the linear semantics.

Definition 13 *Given a logical set $\mathcal{L}'_N$ with $N \in \{U, C, I, T, S\}$, we define:*

1. ***Its symmetric closure*** $\mathcal{L}'^{\equiv}_N$ *by:* $\sigma \in \mathcal{L}'_N \Rightarrow \sigma \in \mathcal{L}'^{\equiv}_N$ *and* $\neg\sigma \in \mathcal{L}'^{\equiv}_N$; $\sigma_i \in \mathcal{L}'^{\equiv}_N\ \forall i \in I \Rightarrow \bigwedge_{i\in I}\sigma_i \in \mathcal{L}'^{\equiv}_N$.
2. ***Its negative closure*** $\mathcal{L}'^{\neg}_N$ *by:* $\sigma \in \mathcal{L}'_N \Rightarrow \neg\sigma \in \mathcal{L}'^{\neg}_N$; $\sigma_i \in \mathcal{L}'^{\neg}_N\ \forall i \in I \Rightarrow \bigwedge_{i\in I}\sigma_i \in \mathcal{L}'^{\neg}_N$.
3. ***Its positive closure*** $\mathcal{L}'^{\surd}_N$ *by:* $\sigma \in \mathcal{L}'_N \Rightarrow \sigma \in \mathcal{L}'^{\surd}_N$; $\sigma_i \in \mathcal{L}'^{\surd}_N\ \forall i \in I \Rightarrow \bigwedge_{i\in I}\sigma_i \in \mathcal{L}'^{\surd}_N$.

Whenever we have a bag of "good" properties (such as $\mathcal{L}'_N$ above), if we want to assert by means of a single formula which is the subset of properties that a certain element satisfies, it is not sufficient to assert that it satisfies each one of them: we also need to assert that it does not satisfy all the rest. This is why we need formulas in the symmetric closure. Instead, if we can only manage formulas from the negative (resp. positive) closure, we can only assert that the element has at most (resp. at least) the enumerated properties. Next we present the unified logics for all the linear semantics in the *enlarged spectrum*.

Definition 14 *Inspired by the orders* $\leq^{l}_N$, $\leq^{l\supseteq}_N$, $\leq^{lf}_N$ *and* $\leq^{lf\supseteq}_N$, *we define the set of formulas* $\mathcal{L}'_{\leq^{l}_N}$, $\mathcal{L}'_{\leq^{l\supseteq}_N}$, $\mathcal{L}'_{\leq^{lf}_N}$ *and* $\mathcal{L}'_{\leq^{lf\supseteq}_N}$, *respectively, by means of the rules:*

1. $\top \in \mathcal{L}'_{\leq^{l}_N}$; $\varphi \in \mathcal{L}'_{\leq^{l}_N}$, $\sigma \in \mathcal{L}'^{\equiv}_N \Rightarrow \sigma\wedge\varphi \in \mathcal{L}'_{\leq^{l}_N}$; $\varphi \in \mathcal{L}'_{\leq^{l}_N}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\leq^{l}_N}$.
2. $\top \in \mathcal{L}'_{\leq^{l\supseteq}_N}$; $\varphi \in \mathcal{L}'_{\leq^{l\supseteq}_N}$, $\sigma \in \mathcal{L}'^{\neg}_N \Rightarrow \sigma\wedge\varphi \in \mathcal{L}'_{\leq^{l\supseteq}_N}$; $\varphi \in \mathcal{L}'_{\leq^{l\supseteq}_N}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\leq^{l\supseteq}_N}$.
3. $\top \in \mathcal{L}'_{\leq^{lf}_N}$; $\sigma \in \mathcal{L}'^{\equiv}_N \Rightarrow \sigma \in \mathcal{L}'_{\leq^{lf}_N}$; $\varphi \in \mathcal{L}'_{\leq^{lf}_N}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\leq^{lf}_N}$.
4. $\top \in \mathcal{L}'_{\leq^{lf\supseteq}_N}$; $\sigma \in \mathcal{L}'^{\neg}_N \Rightarrow \sigma \in \mathcal{L}'_{\leq^{lf\supseteq}_N}$; $\varphi \in \mathcal{L}'_{\leq^{lf\supseteq}_N}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\leq^{lf\supseteq}_N}$.

Note that for the coarsest semantics (e.g. those corresponding to plain refusals and plain readiness when $N = I$) we only observe N at the end of the formula. Instead, the other two logics introduce additional conjunctions that allow to observe N along the computations. Moreover, we have used the negative (resp. symmetric) closure in the "failures based" (resp. "readies based") semantics.

We can use the positive closure to define two new semantics that were not studied in [2, 3] nor elsewhere, as far as we know. They are defined by observing partial offers along a computation, or just at its end. We say that X is a partial offer of p if $X \subseteq I(p)$. It is clear the duality w.r.t. the failures semantics, where F is a failure of p if $I(p) \subseteq \overline{F}$. We can introduce these two new semantics at each layer of the spectrum, by defining the corresponding partial offers for each $N \in \{U, C, I, T, S\}$.

Definition 15 1. *The semantics of* ***partial offer traces*** *for the constraint N is that defined by the logic* $\mathcal{L}'_{\leq^{l\subseteq}_N}$ *with* $\top \in \mathcal{L}'_{\leq^{l\subseteq}_N}$; $\varphi \in \mathcal{L}'_{\leq^{l\subseteq}_N}$, $\sigma \in \mathcal{L}'^{\surd}_N \Rightarrow \sigma\wedge\varphi \in \mathcal{L}'_{\leq^{l\subseteq}_N}$; $\varphi \in \mathcal{L}'_{\leq^{l\subseteq}_N}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\leq^{l\subseteq}_N}$.

2. *The semantics of* ***partial offers*** *for the constraint N is that defined by the logic* $\mathcal{L}'_{\leq^{lf\subseteq}_N}$ *with* $\top \in \mathcal{L}'_{\leq^{lf\subseteq}_N}$; $\sigma \in \mathcal{L}'^{\surd}_N \Rightarrow \sigma \in \mathcal{L}'_{\leq^{lf\subseteq}_N}$; $\varphi \in \mathcal{L}'_{\leq^{lf\subseteq}_N}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\leq^{lf\subseteq}_N}$.

Duality between failures and partial offers causes the picture of the complete layer of linear semantics for each N to become two diamonds that share the side corresponding to the readies-based semantics.

Proposition 5 1. $\mathcal{L}'_F$ *and* $\mathcal{L}'_{\leq^{lf\subseteq}_I}$ *are not comparable:* $p \leq^{lf\supseteq}_I q \not\Rightarrow p \leq^{lf\subseteq}_I q$ *and* $p \leq^{lf\subseteq}_I q \not\Rightarrow p \leq^{lf\supseteq}_I q$.

2. $\mathcal{L}'_{FT}$ *and* $\mathcal{L}'_{\leq^{l\subseteq}_I}$ *are incomparable:* $p \leq^{l\supseteq}_I q \not\Rightarrow p \leq^{l\subseteq}_I q$ *and* $p \leq^{l\subseteq}_I q \not\Rightarrow p \leq^{l\supseteq}_I q$.

Proof. In fact we have a stronger result combining the two statements: if we consider $p = ab + ac$, $q = a(b + c)$ and $r = p + q$, we have that $p \sim_{l\supseteq} r$ but $r \not\leq^{lf\subseteq}_I p$ and $q \sim_{l\subseteq} r$ but $r \not\leq^{lf\supseteq}_I q$.

We could obtain similar counterexamples for $N \in \{T, S\}$. Instead, for $N \in \{U, C\}$, which produce the trace semantics and the complete traces semantics, respectively, it is easy to prove that the six logics of the layer are indeed equivalent.

Proposition 6 *We have (1)* $\mathcal{L}'_{\leq_U^{lf}} = \mathcal{L}'_{\leq_U^{l}} = \mathcal{L}'_{\leq_U^{l\supseteq}} = \mathcal{L}'_{\leq_U^{l\subseteq}} = \mathcal{L}'_{\leq_U^{lf\supseteq}} = \mathcal{L}'_{\leq_U^{lf\subseteq}} = \mathcal{L}_T$ *and (2)* $\mathcal{L}'_{\leq_C^{lf\supseteq}} = \mathcal{L}'_{\leq_C^{lf\subseteq}} = \mathcal{L}'_{\leq_C^{l\supseteq}} = \mathcal{L}'_{\leq_C^{l\subseteq}} = \mathcal{L}'_{\leq_C^{lf}} = \mathcal{L}'_{\leq_C^{l}} = \mathcal{L}_{CT}$.

An interesting result illustrating the genericity of our characterizations concerns one of the finest semantics in the classic spectrum: Possible Future (PF). We find PF in Figure 1 below 2S, probably because the more accurate simulation semantics TS was not (yet) included in the spectrum. This is corrected in the *enlarged spectrum* in Figure 2. Considering $N = T$, we have indeed the following result.

Proposition 7 *We have* $\mathcal{L}'_{\leq_T^{lf}} = \mathcal{L}_{PF}$.

4.3 Logical characterization of the deterministic branching semantics

Next we consider the deterministic branching semantics. In the classic spectrum the only such semantics is *Possible Worlds* (PW), but there is one such semantics for each level of the *enlarged spectrum*.

Definition 16 *For each* $N \in \{U, C, I, T, S\}$*, we define the formulas of* $\mathcal{L}'_{D_N}$ *by:* $\top \in \mathcal{L}'_{D_N}$; $\varphi \in \mathcal{L}'_{D_N}$, $\sigma \in \mathcal{L}'^{\equiv}_N \Rightarrow \sigma \wedge \varphi \in \mathcal{L}'_{D_N}$; $X \subseteq \mathrm{Act}, \varphi_a \in \mathcal{L}'_{D_N} \forall a \in X \Rightarrow \bigwedge_{a\in X} a\varphi_a \in \mathcal{L}'_{D_N}$.

For $N = I$ we obtain the unified logical characterization of the PW semantics.

Proposition 8 *We have* $\mathcal{L}'_{D_I} \supseteq \mathcal{L}_{PW}$.

By the way, $\mathcal{L}'_{D_I}$ and $\mathcal{L}_{PW}$ are not equivalent, but this is caused by the fact that the original logical characterization $\mathcal{L}_{PW}$ was wrong. It can be checked, for instance, that taking $p = abc + a(bc + d) + ab$ and $q = a(bc + d) + ab$ we have $p \nsim_{PW} q$, but $p \sim_{\mathcal{L}_{PW}} q$, since $\mathcal{L}_{PW}$ cannot "observe" the intermediate offer that makes the possible world abc different from those of q. Instead, the formula $\varphi \equiv a(\neg d \wedge bc) \in \mathcal{L}_{D_I}$ is enough to distinguish p and q, since we have $p \models \varphi$ and $q \not\models \varphi$.

Formulas \ Constraints (N)	U	C	I	T	S	B
$\top \in \mathcal{L}'_N$	•	•	•	•	ν	ν
$\neg\top = \bot \in \mathcal{L}'_N$	ν	ν	ν	ν	ν	ν
$\neg 0 \in \mathcal{L}'_N$		•	•	ν	ν	ν
$a \in Act \Rightarrow a\top \in \mathcal{L}'_N$			•	ν	ν	ν
$\varphi \in \mathcal{L}'_N$, $a \in Act \Rightarrow a\varphi \in \mathcal{L}'_N$				•	•	•
$\varphi_i \in \mathcal{L}'_N\ \forall i \in I \Rightarrow \bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_N$					•	•
$\varphi \in \mathcal{L}'_N \Rightarrow \neg\varphi \in \mathcal{L}'_N$						•

Table 2: Logical characterizations of the semantics used as constraints in the N-constrained semantics

In Tables 2 and 3, we present all our results in a three-dimensional way: Table 3 shows the rules defining the logics characterizing each of the semantics at each layer of the *enlarged spectrum* (we provide an additional column with the particularization for $N = I$), while Table 2 contains the logics that characterize the constraint governing each of these "layers". As commented above, there are two semantics that appear in both tables, although disguised with different names: $T = \leq_U^l$ (in fact, it is also equal to the other three linear *U-semantics*) and $S = US$.

Formulas \ Semantics ($\mathcal{Y}_\mathcal{N}$)	$\leq_N^{lf\supseteq}$	$\leq_N^{lf}$	$\leq_N^{l\supseteq}$	$\leq_N^{l}$	D_N	NS	$N \in \{U,C,I,T,S\}$
	F	R	FT	RT	PW	RS	when $N = I$
$\top \in \mathcal{L}'_{\mathcal{Y}_N}$	•	•	•	•	•	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$, $a \in Act \Rightarrow$ $a\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$	•	•	•	•	ν	•	
$\varphi \in \mathcal{L}_N^{\neg} \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$	•	ν	ν	ν	ν	ν	
$\varphi \in \mathcal{L}_N^{\equiv} \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$		•		ν	ν	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$, $\sigma \in \mathcal{L}_N^{\neg} \Rightarrow$ $\sigma \wedge \varphi \in \mathcal{L}'_{\mathcal{Y}_N}$			•	ν	ν	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$, $\sigma \in \mathcal{L}_N^{\equiv} \Rightarrow$ $\sigma \wedge \varphi \in \mathcal{L}'_{\mathcal{Y}_N}$				•	•	ν	
$X \subseteq Act$, $\varphi_a \in \mathcal{L}'_{\mathcal{Y}_N}$ $\forall a \in X \Rightarrow$ $\bigwedge_{a\in X} a\varphi_a \in \mathcal{L}'_{\mathcal{Y}_N}$					•	ν	
$\varphi_i \in \mathcal{L}'_{\mathcal{Y}_N}$ $\forall i \in I \Rightarrow$ $\bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{\mathcal{Y}_N}$						•	
$\varphi \in \mathcal{L}_N \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$						•	
$\varphi \in \mathcal{L}_N \Rightarrow$ $\neg\varphi \in \mathcal{L}'_{\mathcal{Y}_N}$						•	

Table 3: Our new logical characterizations for the semantics at each level of the ltbt-spectrum

5 Relating the unified logics and the unified observational model

In this Section we will relate the unified logical characterizations and the unified observational semantics developed in [3]. As we indicated in Section 2, we have to restrict ourselves to finite image processes to obtain the result. As a byproduct, we get for this kind of processes that the finite parts of each of the corresponding languages, that are obtained by intersection with $\mathcal{L}^f_{HM}$, give us a pure finite logical characterization of the semantics. We start by considering the following concept of normal formula.

Definition 17 (Normal formula $\mathcal{N}(\mathcal{L})$) *1. Given a set of formulas $\mathcal{L}$, whose outermost operator is not the conjunction, we define the set of induced normal formulas, $\mathcal{N}(\mathcal{L})$, starting with $\top$ and adding those formulas that can be generated by applying the clause: If $\Gamma_1, \Gamma_2 \subseteq \mathcal{L}$, $\{a_i \mid i \in I\} \subseteq Act$ and $\varphi_i \in \mathcal{N}(\mathcal{L})$, then $(\bigwedge_{\sigma\in\Gamma_1} \sigma \wedge \bigwedge_{\sigma\in\Gamma_2} \neg\sigma) \wedge \bigwedge_{i\in I} a_i\varphi_i \in \mathcal{N}(\mathcal{L})$.*

2. Now, for each $N \in \{U,C,I,T,S\}$ and each $\mathcal{Y}_\mathcal{N} \in \{NS, \leq_N^l, \leq_N^{l\supseteq}, \leq_N^{lf}, \leq_N^{lf\supseteq}, \leq_N^{l\subseteq}, \leq_N^{lf\subseteq}, D_N\}$ in the spectrum, we define the set of normal formulas, $\mathcal{N}_{\mathcal{Y}_N}(\mathcal{L}''_\mathcal{N}) \subseteq \mathcal{L}'_{\mathcal{Y}_N}$ simply as: $\mathcal{N}_{\mathcal{Y}_N}(\mathcal{L}''_\mathcal{N}) = \mathcal{N}(\mathcal{L}_\mathcal{N}) \bigcap \mathcal{L}'_{\mathcal{Y}_N}$ where $\mathcal{L}''_N$ is the set of formulas in $\mathcal{L}'_N$ whose outermost operator is not the conjunction.

Remark 3 *First note that the clause in Def. 17.1 is a bit complicated: initially, we can apply it starting with $I = \emptyset$, and in this way we can obtain the first (non-trivial) normal formulas; then we can apply it recursively to obtain new, more complex, normal formulas; instead, the formulas in the two first subformulas come always from the original set $\mathcal{L}$. Also note that we admit the use of infinite conjunction in those two first subformulas. As a consequence, these formulas could also have infinite depth (as infinite formulas in (the infinite generalizations of) $\mathcal{L}_{HM}$). However, if we define the normal depth of formulas in $\mathcal{N}(\mathcal{L}_\mathcal{N})$ as that obtained by counting the recursive nesting in the application of Def. 17, then any normal formula has finite normal depth, and the set they form can be explored by structural induction.*

Theorem 4 *Each set of normal formulas $\mathcal{N}_{\mathcal{Y}_N}(\mathcal{L}''_\mathcal{N})$ associated to each of the semantics in the spectrum is equivalent to the full set of formulas $\mathcal{L}'_{\mathcal{Y}_N}$.*

Definition 18 *We define the set of complete normal formulas $CN(\mathcal{L})$ (resp. the set of complete normal formulas associated to each semantics in the spectrum, $CN_{\mathcal{Y}_N}(\mathcal{L}''_N)$) as the set of normal formulas (resp. the set of normal formulas associated to each semantics in the spectrum) that satisfy the condition $\Gamma_2 = \overline{\Gamma_1}$, whenever the rule in Def. 17 is applied in the generation of each formula.*

Next we state that infinite conjunction in Def. 17 can be approximated by finite conjunction.

Theorem 5 *If we restrict ourselves to finite image processes, any complete normal formula $\varphi \in CN(\mathcal{L})$ can be approximated by a set of finite normal formulas $\{\varphi^k \mid k \in \mathbb{N}\}$ that only use finite conjunction, that is, we have $p \models \varphi \Leftrightarrow p \models \varphi^k \ \forall k \in \mathbb{N}$.*

Theorem 6 *We can define a natural correspondence between the set of complete normal formulas associated to a semantics $CN_{\mathcal{Y}_N}(\mathcal{L}''_N)$ and the corresponding domain of observations BGO_N or LGO_N. That correspondence $\leftrightarrow$ satisfies that $\varphi \leftrightarrow \theta \Rightarrow (p \models \varphi \Leftrightarrow \theta \in XGO_N(p))$ with $X = B$ or $X = L$. Moreover, this correspondence produces the following results for each of the semantics in the spectrum:*

1. *The set of complete normal formulas $CN_{NS}(\mathcal{L}''_N)$ (resp. $CN_{\mathcal{D}_N}(\mathcal{L}''_N)$) and the domain of branching general observations GBO_N (resp. $dBGO_N$) are isomorphic, that is, $\leftrightarrow$ is one to one.*
2. *The set of complete normal formulas $CN_{\leq^l_N}(\mathcal{L}''_N)$, $CN_{\leq^{l\supseteq}_N}(\mathcal{L}''_N)$ and the domain of linear general observations LGO_N are isomorphic, that is, $\leftrightarrow$ is one to one.*
3. *The set of complete normal formulas $CN_{\leq^{lf}_N}(\mathcal{L}''_N)$ (resp. $CN_{\leq^{lf\supseteq}_N}(\mathcal{L}''_N)$) and the quotient domain $LGO_N/_{\simeq^{lf}_N}$ (resp. $LGO_N/_{\simeq^{lf\supseteq}_N}$) are isomorphic, that is, $\leftrightarrow^{-1}$ is injective and $\varphi \leftrightarrow \theta$ iff $\theta \simeq^{lf\supseteq}_N \theta_\varphi$, for some adequate θ_φ.*

Theorem 7 *The logical semantics $\sqsubseteq'_{\mathcal{Y}_N}$ induced by the logic $\mathcal{L}'_{\mathcal{Y}_N}$, where $\mathcal{Y}_N \in \{NS, \leq^l_N, \leq^{l\supseteq}_N, \leq^{lf}_N, \leq^{lf\supseteq}_N, D_N\}$, is equivalent to the corresponding observational semantics, defined at Def. 6 and Def. 7.*

6 The real diamond structure

Now we will explore in more detail the real structure of the extended spectrum, as it was already done at [2]. One could think that each diamond in that spectrum corresponds to a lattice structure. However, this is not the case: there is another semantics coarser than both N-readiness and N-failure traces and finer than N-failures, and another finer than those two semantics and coarser than N-ready traces.

Figure 4: The diamond below ready simulation

Focusing on the case $N = I$ the obtained complete structure is that shown in Figure 4, in which we include the new join semantics $R \wedge FT$ and the meet one $R \vee FT$. As proved in [3], the meet semantics $R \vee FT$ was already studied by Roscoe under the name of revivals semantics in [11].

Since Readiness semantics observes the ready set at the end of the trace, while Failure Traces observes failures during the computation, it is natural to expect that the join semantics $R \wedge FT$ will observe both failures during the computation and ready sets at the end. This is indeed the case. The corresponding observational characterization in the general case is obtained by means of a new order $\leq^{l\supseteq\wedge f}_N$ on LGO_N.

Definition 19 *Let $\zeta,\zeta' \subseteq LGO_N$, we define*

$$\zeta \leq_N^{l\supseteq\wedge f} \zeta' \Leftrightarrow \forall\, X_0a_1X_1\ldots X_n \in \zeta\ \exists\, Y_0a_1Y_1\ldots Y_n \in \zeta'\ (\forall i \in 0..n-1\ X_i \supseteq Y_i) \wedge X_n = Y_n\ .$$

It is easy to see that $\leq_N^{l\supseteq\wedge f}$ is indeed the conjunction of $\leq_N^{l\supseteq}$ and $\leq_N^{lf}$, that is, $\zeta \leq_N^{l\supseteq\wedge f} \zeta' \Leftrightarrow \zeta \leq_N^{l\supseteq} \zeta' \wedge \zeta \leq_N^{lf} \zeta'$. The observational characterization of the meet semantics $R \vee FT$ is a bit more complicated.

Definition 20 *Let $\zeta,\zeta' \subseteq LGO_N$, we define*

$$\zeta \leq_N^{l\supseteq\vee f} \zeta' \Leftrightarrow \forall\, X_0a_1X_1\ldots X_n \in \zeta\ \exists\, \{Y_0a_1Y_1\ldots Y_n^j | j \in J\} \subseteq \zeta' \textit{ such that } X_n = \textstyle\bigcup_{j\in J} Y_n^j\ .$$

By means of some simple algebraic manipulations we can get the following equivalent expression:

$$\zeta \leq_N^{l\supseteq\vee f} \zeta' \Leftrightarrow \forall\, X_0a_1X_1\ldots X_n \in \zeta\ \forall a \in X_n\ \exists\, Y_0a_1Y_1\ldots Y_n \in \zeta' \text{ such that } (a \in Y_n \wedge Y_n \subseteq X_n)\ .$$

Next we present the logical characterizations of these new semantics. Obviously, they are in the linear side of the spectrum and therefore they will have a similar structure to those for the linear semantics studied before. Once again, we start with the particular case $N = I$. $R \wedge FT$ is finer than both R and FT, and the logic characterizing it will be just the union of those characterizing R and FT. In the case of $R \vee FT$ we need to connect the clauses that define those two logics in an adequate way.

Definition 21

1. *We define the set of formulas $\mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$, as that generated by the clauses: $\top \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$; $\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$, $\sigma \in \mathcal{L}_I^{\neg} \Rightarrow \sigma\wedge\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$; $\sigma \in \mathcal{L}_I^{\equiv} \Rightarrow \sigma \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$; $\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$.*
2. *We define the set of formulas $\mathcal{L}'_{\leq_I^{l\supseteq\vee f}}$ as that generated by the clauses: $\top \in \mathcal{L}'_{\leq_I^{l\supseteq\vee f}}$; $\sigma,\sigma_j \in \mathcal{L}'_I \forall\, j \in J \Rightarrow (\sigma \wedge \bigwedge_{j\in J} \neg\sigma_j\top) \in \mathcal{L}_{\leq_I^{l\supseteq\vee f}}$; $\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\vee f}}$, $a \in \mathrm{Act} \Rightarrow a\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\vee f}}$.*

Example 2 *P_2 and P_3 in Figure 3 satisfy $P_2 \sim_F P_3$, but $P_2 \not\leq_{R\vee F} P_3$. Taking p= abc+a(bd+c) and q= p+ a(bc+c) we have $p \sim_{R\wedge FT} q$ but $p \not\sim_{RT} q$.*

Theorem 8 *The logical semantics $\sqsubseteq'_{\leq_I^{l\supseteq\wedge f}}$ (resp. $\sqsubseteq'_{\leq_I^{l\supseteq\vee f}}$) induced by the logic $\mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$ (resp. $\mathcal{L}'_{\leq_I^{l\supseteq\vee f}}$) is equivalent to the observational semantics defined by LGO_I, with the order $\leq_I^{l\supseteq\wedge f}$ (resp. $\leq_I^{l\supseteq\vee f}$.)*

Proof. In the case of $R \wedge FT$ we just need to check that $\mathcal{L}'_{\leq_I^{l\supseteq\wedge f}} = \mathcal{L}'_{\leq_I^{l\supseteq}} \cup \mathcal{L}'_{\leq_I^{lf}}$. The meet of two semantics is not always defined by the intersection of the corresponding logics. However, in this case we have that $\mathcal{L}'_{\leq_I^{l\supseteq\vee f}} = \mathcal{L}'_{\leq_I^{l\supseteq}} \cap \mathcal{L}'_{\leq_I^{lf}}$, and then to check that it defines $R \vee FT$ it is enough to see that $p \not\sqsubseteq_I^{l\supseteq\vee f} q \Rightarrow (\exists\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\vee f}}\ p \models \varphi \wedge q \not\models \varphi)$, which is nearly immediate.

By replacing I above by the generic N we get the definitions and results for the general case.

7 Conclusions and future work

We have concluded in this paper the work on unification of all the strong process semantics by considering here the logic approach, while [2, 3] considered the observational and the equational approaches. As in the previous cases, our main goal was to clarify the relationships between all the process semantics, that were classified in a slightly messy way in [6]. Our starting point has been the Hennessy-Milner Logic [8]: we have looked for sublogics with a simple structure, that characterize each of the semantics in the *enlarged spectrum*. The difference between branching-time semantics and linear-time semantics is the key point to isolate the ingredients that, combined in different ways, produce the different semantics.

It is interesting to comment on the difference between the observational and the logical characterizations. Note that in the observational framework the observations had a complex structure, where local observations informed us about the (static) properties of the states of a process, while the arcs gave us the dynamic information. Instead, the formulas of the logic *HML* do not possess of such structure, having only a *low level* structure induced by the combination of prefix and conjunction. This is why we needed to introduce normal forms in order to build the *high level* structure of observations at the formulas.

Came as a surprise to us the discovery of two more linear semantics at each layer of the spectrum. Moreover, we found out that the classic logical characterization of Possible Worlds (PW) was wrong. A too ad-hoc selection of the rules defining each logic was probably the cause, that we discovered when trying to unfold the original characterization to look for the equivalent presentation inside our model.

Now that we have available all the unified characterizations of the semantics we have a much clearer picture of the spectrum, and we can use the parameterized definitions to prove generic properties of all or a part of the semantics in a generic way, without having to repeat similar proofs for each of them.

There are several directions in which we plan to extend our work. Weak semantics are an obvious target: if there are indeed many strong process semantics, once we introduce internal actions a terrible explosion occurs [5], and the unification work is even more necessary in order to clarify which are the most interesting semantics and what the differences between them are. Another interesting direction comes from the combinations of logic and algebra, as done by Luttgen and Vogler [10, 9]. Again, we are interested in studying whether their proposal is canonical or can be parameterized in some way in order to obtain other interesting combinations. Finally, a couple of papers [1, 7] have appeared recently, where the logical characterizations of the non-interleaving semantics are developed.

References

[1] P. Baldan & S. Crafa (2010): *A Logic for True Concurrency*. In: *CONCUR 2010*, *LNCS* 6269, Springer, pp. 147–161, doi:10.1007/978-3-642-15375-4_11.

[2] D. de Frutos, C. Gregorio & M. Palomino (2009): *On the unification of process semantics: equational semantics*. *ENTCS* 249, pp. 243–267, doi:10.1016/j.entcs.2009.07.093.

[3] D. de Frutos, C. Gregorio & M. Palomino (2009): *On the unification of process semantics: observational semantics*. In: *SOFSEM 2009*, *LNCS* 5404, Springer, pp. 279–290, doi:10.1007/978-3-540-95891-8_27.

[4] D. de Frutos-Escrig & C. Gregorio-Rodríguez (2008): *Universal Coinductive Characterisations of Process Semantics*. In: *IFIP TCS 2008*, pp. 397–412, doi:10.1007/978-0-387-09680-3_27.

[5] R. J. van Glabbeek (1993): *The Linear Time - Branching Time Spectrum II*. In: *CONCUR 1993*, *LNCS* 715, Springer, pp. 66–81, doi:10.1007/3-540-57208-2_6.

[6] R.J. van Glabbeek (2001): *The linear time-branching time spectrum* I*: the semantics of concrete, sequential processes*. In J.A. Bergstra, A. Ponse & S.A. Smolka, eds.: *Handbook of Process Algebra*, Elsevier, pp. 3–99.

[7] J. Gutierrez (2009): *Logics and Bisimulation Games for Concurrency, Causality and Conflict*. In: *FOSSACS 2009*, *LNCS* 5504, Springer, pp. 48–62, doi:10.1007/978-3-642-00596-1_5.

[8] M. Hennessy & R. Milner (1985): *Algebraic laws for nondeterminism and concurrency*. *Journal of the ACM* 32, pp. 137–161, doi:10.1145/2455.2460.

[9] G. Lüttgen & W. Vogler (2009): *Safe Reasoning with Logic LTS*. In: *SOFSEM 2009*, *LNCS* 5404, Springer, pp. 376–387, doi:10.1007/978-3-540-95891-8_35.

[10] G. Lüttgen & W. Vogler (2010): *Ready simulation for concurrency: It's logical!* *Inf. Comput.* 208(7), pp. 845–867, doi:10.1016/j.ic.2010.02.001.

[11] A. W. Roscoe (2009): *Revivals, stuckness and the hierarchy of CSP models*. *J. Log. Algebr. Program.* 78(3), pp. 163–190, doi:10.1016/j.jlap.2008.10.002.

5.2 A unifying theory for the characterization of process semantics

Once we were able to logically characterize the different process semantics in a unified way, we worked together with Profs. C. Gregorio Rodríguez and M. Palomino Tarjuelo to produce the technical and nearly encyclopedic article which collects our unification theories for process semantics. Although most of the work was the result of the previous research conducted by my supervisor and Prof. C. Gregorio Rodríguez, my personal involvement focussed on all the contents related with the logic framework that I had investigated. The presentation, together with the other approaches, produced a richer vision. In particular, the relations between the logic framework and the observational and axiomatic frameworks, were stressed. The logical approach is dual to the axiomatic one by a Galois connection, so that whenever the former produces more complex characterizations, the other is simpler, and vice-versa.

The journal *Logical Methods in Computer Science*, which is probably the most prestigious open access journal in the field of Theoretical Computer Science, was chosen to publish our work. With it we considered concluded, at least for a time, our research on the field of unification of process semantics. Next, it is certainly a pleasure to conclude this short introduction quoting the highly positive review of this paper done by Gerald Lüttgen in Mathematical Reviews (*nr. 3078096*); it seems that we did a quite interesting job.

> This comprehensive article considers an expanded version of van Glabbeek's original linear-time branching-time spectrum for strong process semantics: bisimulation, 2-nested bisimulation, ready simulation, complete simulation, simulation, possible futures, impossible futures, possible worlds, ready trace, failure trace, readiness, failures, completed trace and trace semantics.
>
> The simulation semantics are employed as a driver for classifying the remaining semantics, as is evident from the authors' unified observational characterization of process semantics: each simulation is characterized by a branching observation and a local observation function, and gives rise to a corresponding version for each of the traditional linear semantics (failures, readiness, failure trace and ready trace). This insight leads, in a systematic way, to new axiomatizations of all semantics in the spectrum, which are based on two parametric axiom patterns only. It also allows for generic derivations of logical characterizations in the style of Hennessy and Milner. Finally, the article

presents a unified operational characterization, which can, however, be seen as slightly *ad hoc*.

The authors demonstrate the utility of their unifying theory by characterizing Roscoe's stable revivals semantics in terms of their observational framework. They also identify two new semantics as the least upper bound and, respectively, the largest lower bound of readiness and failure trace semantics, where one is finer than failures and the other coarser than ready trace semantics.

The well-written and self-contained article is a highly recommended read: the authors' unifying approach does not only nicely present known results on behavioral equivalences and preorders, but it also enables the reader to gain new insight through the distinguished ways of deriving the article's results.

Logical Methods in Computer Science
Vol. 9(2:11)2013, pp. 1–74
www.lmcs-online.org

Submitted Mar. 6, 2012
Published Jun. 26, 2013

UNIFYING THE LINEAR TIME-BRANCHING TIME SPECTRUM OF STRONG PROCESS SEMANTICS

DAVID DE FRUTOS ESCRIG [a], CARLOS GREGORIO RODRÍGUEZ [b], MIGUEL PALOMINO [c], AND DAVID ROMERO HERNÁNDEZ [d]

[a,b,c,d] Departamento de Sistemas Informáticos y Computación, Universidad Complutense de Madrid
e-mail address: {defrutos, cgr, miguelpt}@sip.ucm.es, dromeroh@pdi.ucm.es

Abstract. Van Glabbeek's linear time-branching time spectrum is one of the most relevant work on comparative study on process semantics, in which semantics are partially ordered by their discrimination power. In this paper we bring forward a refinement of this classification and show how the process semantics can be dealt with in a uniform way: based on the very natural concept of constrained simulation we show how we can classify the spectrum in layers; for the families lying in the same layer we show how to obtain in a generic way equational, observational, logical and operational characterizations; relations among layers are also very natural and differences just stem from the constraint imposed on the simulations that rule the layers. Our methodology also shows how to achieve a uniform treatment of semantic preorders and equivalences.

2012 ACM CCS: [**Theory of computation**]: Formal languages and automata theory—Formalisms—Algebraic language theory; Formal languages and automata theory—Semantics and reasoning—Program semantics; Logic; Models of computation—Concurrency.

Key words and phrases: process semantics, linear time-branching time spectrum, algebraic languages, simulation semantics, linear semantics, constrained simulation, axiomatizations, unification.

[a,c,d] David de Frutos Escrig, Miguel Palomino and David Romero Hernández were partially supported by the Spanish MEC project DESAFIOS10 TIN2009-14599-C03-01 and the project PROMETIDOS S2009/TIC-1465.

[b] Carlos Gregorio Rodríguez was partially supported by the Spanish MEC project ESTuDIo TIN2012-36812-C02-01.

LOGICAL METHODS IN COMPUTER SCIENCE DOI:10.2168/LMCS-9(2:11)2013

Contents

1. Introduction

Since the foundational work by Robin Milner [41, 42] and Tony Hoare [32] on process semantics, there has been a multitude of proposals to endow processes with meaning and to define equivalence and preorder relations over them. Among the most relevant work are those of Matthew Hennessy [30], who introduced the testing methodology defining process semantics from test cases, and those of Jan Bergstra and Jan Willen Klop [11], later continued by Jos Baeten and Peter Weijland [10], which were based on an axiomatic approach.

These proposals define algebraic languages for the specification of processes, diverging in subtle details concerning the treatment of non-determinism and parallelism. These aspects are captured by means of certain operators which may (strongly) vary in each particular language.

Focusing on equivalences, it is interesting to note how the pioneering work in this area already established two fundamental notions, bisimulation and traces/failures, that constitute an upper and a lower bound on the natural framework in which other process equivalences can be studied. Hoare—with his characteristic clarity—summarizes the situation in the following paragraph.

> *CCS makes many distinctions between processes which would be regarded as identical in this book. The reason for this is that CCS is intended to serve as a framework for a family of models, each of which may make more identifications than CCS but cannot make less. To avoid restricting the range of models, CCS makes only those identifications which seem absolutely essential. In the mathematical model of this book [CSP] we have pursued exactly the opposite goal —we have made as many identifications as possible, preserving only the most essential distinctions.* [32]

In between these two fundamental notions of equivalence—bisimulation and traces—the last two decades of the 20th century witnessed the surge of a large variety of new equivalences associated to new calculi and process algebras, whose aim was to explore the different needs for expressivity and distinction capabilities in many applications.

The most important taxonomic work on process semantics was carried out by Rob van Glabbeek as part of his doctoral dissertation [54]. In two papers, titled *Linear time-branching time spectrum* [55, 56], he collected the most important of these equivalences establishing, among other results that we will comment on, a classification based on their capability to distinguish processes. The first of the papers concentrate on *strong* semantics, in the sense that they consider each action processes perform as being observable by their environment. The second paper consider the inclusion in the language of a new and invisible action τ; process semantics considering this internal action are usually called *weak* semantics. Figure 1 shows a slightly expanded version of the spectrum proposed by van Glabbeek for the case of strong semantics in [55]. These strong semantics, that do not consider at all the special role of internal actions τ, are the only ones that we consider in this paper.

This array of semantics is supported by many authors who claim that there is no single "good" definition. Process theory can be applied in a wide spectrum of contexts and situations and the concrete uses will have a decisive influence in the election of what a suitable semantics should be.

> *The choice of a suitable semantics may depend on the tools an environment has, to distinguish between certain processes. It is conceivable that a concurrency theory is equipped with different semantics, and has the capacity to express equality on different levels.* [57]

The possibility to define several and varied semantics can then be considered to be an advantage of the theory, since it allows for the necessary flexibility to reflect different notions of processes and equivalence and preorder relations over them.

Nevertheless, this multiplicity has gone hand in hand in the literature with an individual study of each of the semantics that somehow makes the whole theory less appealing because such a cornucopia can become a handicap both for its study and its practical application. For instance, although most of the semantic notions defined for processes simultaneously induce both a (pre)order and an equivalence, [1] the literature has frequently overlooked the

[1] A remarkable exception, however, is the bisimulation notion, for which no non-trivial order relation is known.

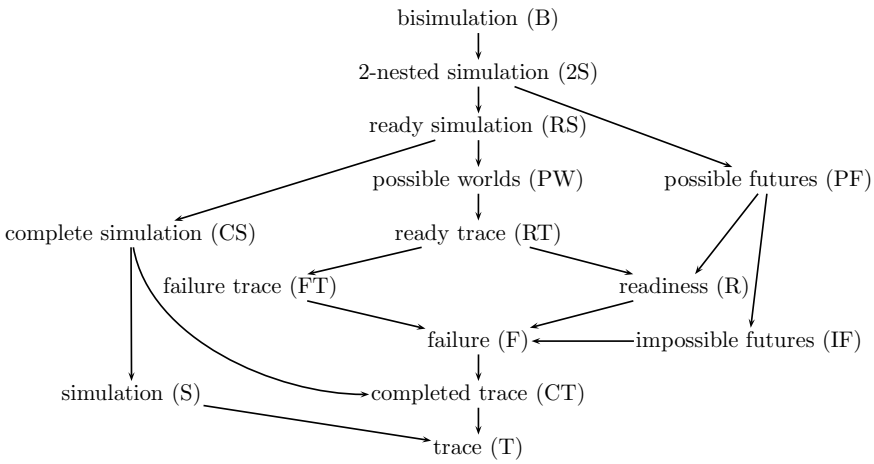


Figure 1: Linear time-branching time spectrum.

fact that these two notions are mutually intertwined, as we will show later. Likewise, the study of the concrete models has been usually undertaken paying little attention to the other semantics or to the relations among them, even though it is well-known that there exist "families" of semantics—such as the linear semantics—which are undoubtedly related.

A unified study of semantics has both methodological and practical implications that have been explored along the last years by the authors of this work, for example in [22, 24, 25, 20, 21], and also in work by important researchers in the area [5, 4, 16, 40]. This research shows that a unified view of process semantics is indeed possible.

This is precisely the main goal we set to reach with our work: we aim to study process semantics in a generic way, making the equivalence and (pre)order relations our object of study in order to find patterns, to identify families, to search for properties among these relations, so that we obtain generic results that need not be proved again and again for each of the semantics.

We aim, in a nutshell, at a unifying view of process semantics that can be used to understand them both jointly and individually and that allows to continue with their theoretical study in a more focused manner, helping to identify those properties a semantics should have for a particular application.

1.1. **Overview of results.** This paper contains a consolidated and extended presentation of the unification of observational, equational and the logic process semantics published in [20, 19, 47] for strong behavioral semantics.[2] We take advantage of the joint and larger presentation of the subject to tighten the connections between the different views. Besides, we make the paper mostly self-contained providing proofs for all the results; we also complete the study with new results not included in [20, 19, 47]. We have also completed the

[2]We comment in Section 10 the work of some of the authors on unification of weak semantics.

revision of the unification of strong process semantics with a section devoted to the unified presentation of the operational semantics.

Next we describe the main results we have obtained. They can be used as a roadmap for reading the paper and to understand the technical details in the following sections.

- One of the most generic results we have proved is the existence of two essential families of semantics: branching semantics and linear semantics. Certainly, this was already hinted by van Glabbeek when he named its spectrum of semantics "linear time-branching time".

  Our results show that the most representative branching semantics have characterizations as simulation semantics. Moreover, every simulation has a natural family of coarser linear semantics associated to it that inherit some of its properties.

  In Figure 11 (page 34), branching semantics are located to the left and each of them defines a layer of "induced" linear semantics, to their right. For example, ready simulation is the branching semantics from which the classic diamond of linear semantics composed of failures, readiness, failure trace, and ready trace semantics is generated. These semantics, as we will later see in detail, inherit some of their axiomatic characterizations directly from the ready simulation. In addition, the same layer also contains the possible worlds semantics, which is a deterministic branching semantics.

  Even though the axiomatic characterizations contained in Section 3 already show this dependency between linear and branching semantics—see Figure 5 (page 17)—it is in Section 4 where, using techniques from denotational semantics, the relations between the original branching semantics and the induced linear semantics can be fully appreciated. The relationships among the different linear semantics in the same branching layer are also completely specified in that section.
- Equational characterizations reveal in a very concise manner the basic properties of the different semantics. As a result of our research, we have been able to derive a generic axiomatic characterization of the semantics in the spectrum which shows clearly the relationships among them: the uniformity in the definitions of the branching semantics and the different families of linear semantics becomes apparent, as well as the tight relation between each branching semantics and its associated linear semantics. In Section 3 we present all the details related to these axiomatic characterizations.
- Another result that we consider important is that it is indeed possible to establish a clear relationship between the preorder and the equivalence associated to a given semantics. From an axiomatic point of view, in Section 3 we show how these characterizations are closely related. In fact, there are algorithms that allow to easily obtain the axioms for the equivalence from those of the preorder [4, 21], and also the other way around, the axioms for the preorder from those of the equivalence [23, 24].
- We also offer a unifying view of the process semantics based on observational (denotational) semantics, according to which we have classified the process semantics in four categories:
  - bisimulation semantics, which is the finest semantics in the spectrum and the only one that cannot be defined by means of a non-trivial preorder;
  - the simulation semantics (simulation, complete simulation, ready simulation, nested simulation, ...) which are characterized by means of branching observations, that is, labeled trees;
  - the linear semantics (traces, failures, readiness, ...), characterized by linear observations, a degenerated case of branching observations;

– the deterministic branching semantics corresponding to an intermediate class between branching and linear, where observations are deterministic trees. Possible worlds semantics is the only semantics in the original van Glabbeek's spectrum in this class.

Besides their linear or branching nature, semantics are characterized by a local observation function that generates the local observations at the states. For the linear case there is also the possibility of observing this local information in a partial way and this is how for each local observer, in principle, up to four different semantics can be obtained. In particular, this gives rise to the classic diamond below the ready simulation semantics formed by failures, failure trace, readiness, and ready trace semantics.

The uniform presentation of the process semantics that we offer in Section 4 clarifies the relationships and hierarchies among all the semantics; moreover, it will make possible the development of generic proofs of their common properties.

- We also present a unified view of the logical semantics. Again, the bisimulation semantics, which is characterized by the Hennessy-Milner logic HML [31], is our starting point, and then we aim for the sublogics that characterize each of the semantics in the spectrum. Guided by our main unification goal, we have not tried to obtain the smallest possible sets of formulas, but have veered for the largest sublogics that characterize each of the semantics.

  Hence, the finest semantics are characterized by the largest sublogics and in fact we obtained a uniform characterization that informs us about the hierarchy of semantics, by proving that a semantics S_1 is finer than another S_2 if and only if the corresponding logics satisfy $\mathcal{L}_{S_2} \subseteq \mathcal{L}_{S_1}$. Moreover, the classification into branching and linear time semantics is also reflected in the structural definition of each logic. In particular, the branching semantics are characterized by the free use of negation over the formulas that define the corresponding constraint, while the linear semantics at each layer of the spectrum introduce ever more limitations in the subformulas.
- Finally, we also discuss an operational-like presentation of the semantics in the spectrum; more precisely, we consider an evaluation semantics to derive the appropriate data which characterize them. Those data are quite similar to the ones employed for our observational semantics so that it is not them, but the way in which they are derived, that enhances our understanding of the features of each of the semantics and the relationships between them. These presentations somehow generalize the work by Cleaveland and Hennessy [17] on the characterization of the Testing semantics by means of bisimulation. There is also a clear connection with the work by two of the authors of this paper on (Bi)simulation up-to [25].
- A concomitant, but still important result of our work, is of methodological nature: the semantics are amenable to a working methodology that allows for general results that can be applied to families of semantics as well as to yet to be defined semantics. The requirements we impose on these new semantics are relatively mild. An example of this is shown seen in Section 8, where some new process semantics—indeed two new families—smoothly integrate into our general theory. In fact, it was nice to discover that one of them is just the revivals semantics, which has been recently developed by Bill Roscoe [48].
- Each of the characterization frameworks—equational, observational, logical or operational—sheds light on the spectrum in different and complementary ways. This has provided us with different ways to study all the previously known semantics and the relationships between them. That complementary nature also sprang up along our unification work when we discovered one by one all the factors that contribute to the structure of the

extended spectrum. In particular, when considering simple and natural combinations of axioms we found out the new meet semantics in Section 8, while their dual join semantics was discovered in a natural way when considering the observational characterizations. Finally, the semantics of minimal readies (in Section 6.2.2) appeared when investigating the logical framework. While not too important on its own, our unification work has also revealed a mistake in the classic logical characterization of one of the semantics in the original spectrum (see Section 6.2.3): it was the general and systematic approach that guides our uniform characterization that allowed it.

1.2. **Some related work.** Naturally, the goal of defining a global or general theory of process semantics has been around for a long time and several relevant authors in the field have already paved the way that we now tread.

Despite the methodological differences between Milner's work, based on bisimulation, and Hoare's, on denotational semantics, both of them had in common the search for characterizations—logical, axiomatic, observational—that could shed light from different angles on the world of process semantics.

Hennessy introduced the testing methodology to endow processes with semantics, making the notion of equivalence to spring from the application of the interaction principles for processes expressed within the model. Perhaps one of the most important contributions of his work was what he called the "trinity": processes can be seen as syntactic terms in an algebra, as operational descriptions in labeled transition systems, or as denotational objects in a mathematical model. With our work we have somehow extended this trinity in a generic manner to all the semantics in the extended spectrum.

Van Glabbeek's work, *the linear time-branching time spectrum* aimed at the comparison of most known semantics—at the time he developed his seminal work—by presenting them within common frameworks that would allow a comparative study of their properties. Besides providing uniform definitions over transition systems, van Glabbeek also proposed to characterize the semantics in terms of logical formulas. The set of modal formulas whose satisfaction equivalence identifies the same processes as the corresponding semantics is defined. Because of the compositional definition of the corresponding sets of formulas, this characterization can be considered to be denotational semantics.

Another characterization provided by van Glabbeek is the axiomatic one, for which he defines the BCCSP language that is used in this work (see Definition 2.1). Twelve of the semantics in the spectrum are characterized by means of sets of axioms over syntactic terms for this language. For most of them—except for bisimulation, that has no associated order—their characterizations are actually twofold: on the one hand, the natural order relation that defines each semantics—Table 1—and, on the other hand, the induced equivalence—Table 2. Many of these characterizations were previously known but, again, their uniform presentation is one of its main merits.

A deep study—individual as well as comparative—of these axiomatizations and the quest for answers to the new questions that arise from this study has been one of the leading forces behind our research. Actually, some of our most relevant results can be combined into a new way of presenting the spectrum—Figure 11 (page 34)—that allows for a better comprehension of the semantics since it clarifies their relative positions within it and shows the existence of "gaps" that correspond to new semantics whose addition to the graph reflects a desirable regularity that makes it clearer. Hoare's work on the unification of the

study of process algebras [33] was also an important influence. Specially, the relationship between similarity and trace refinement, which we have generalized by establishing the connection between branching time and linear time semantics, and the connection between the denotational, the algebraic, and the operational styles proposed by him and He Jifeng in [34].

As already mentioned, Roscoe has contributed in an independent research effort in parallel with ours to the study of new process semantics by proposing his stable revivals model [48]. He relates his new semantics with other well known linear time semantics and the rediscovery of that semantics in our expanded spectrum gave us the opportunity to present those relationships with a unified and generic light.

There is other relevant work in the area of process theory that has inspired us. The number of contributions is too large to cite all of them here. Anyone interested on finding a more exhaustive list of relevant references may collect them, for instance, from [28, 7, 1, 57, 50]. There the historic evolution of the area and many of the most important contributions to it are reviewed. To them we can add four recent books on process algebras and related subjects [8, 49, 6, 51], presenting different points of view and some of the semantics studied in this paper. Finally, in our Conclusions, we will discuss a bit the work on the generic study and classification of the weak semantics.

1.3. **Paper structure.** We have structured this paper as follows. Section 2 introduces all the basic definitions and notation to properly follow the developments in the following sections.

In Section 3 we propose alternative characterizations for the axiomatizations of the semantics in the spectrum, both for orders and equivalences. All these axiomatizations are based on just two parametrical axiom skeletons that clearly highlight the relations among the different semantics.

Section 4 presents a unified observational characterization for process semantics. One of the key ideas is that constrained simulations are uniformly characterized by a branching observation plus a local observation function. From the observations of a given constrained simulation, the linear semantics in its layer are uniformly derived.

In Section 5 we prove that the equations we presented in Section 3 are deduced from the observations defined in Section 4 in a general way, without using at all the already known axiomatizations for the semantics.

Section 6 follows the trails of Sections 3 and 4 by introducing a unified logical characterization of process semantics.

In Section 7 we prove that the observational characterizations developed in Section 4 allow for generic proofs for the logical characterizations presented in Section 6. Therefore the "trinity" of equations, observations, and logical formulas is established in a generic way for large families of process semantics.

Section 8 is a practical proof of the applicability of our unification proposal. Some new process semantics, that were not listed in the original linear time-branching time spectrum, are easily accommodated in our framework thus getting the corresponding semantic characterizations that we have presented in previous sections.

In Section 9 we conclude the unified presentation of the semantics in the spectrum by developing an operational characterization which mainly produces the information provided by the observational semantics, but inferred in an operational way, using a (unified) set of SOS-like rules.

Finally, in Section 10 we offer some conclusions and lines for future work.

Acknowledgments. We gratefully acknowledge three anonymous referees for their very thoughtful and detailed comments on a previous version of this work, that have greatly helped us to improve the presentation of this material.

2. Preliminaries

Although the main results in this paper are also valid for infinite processes—as we showed in [22, 25]—in order to simplify the presentation of the concepts, we will mainly consider finite processes generated by the basic process algebra BCCSP which contains only the basic process algebraic operators from CCS [42], and CSP [32], but is sufficiently powerful to express all finite synchronization trees [41]. This language has repeatedly been used in unification work, e.g. [4, 58].

Definition 2.1. Given a set of actions Act, the set BCCSP(Act) of processes is defined by the following BNF-grammar:

$$p ::= \mathbf{0} \mid ap \mid p + q$$

where $a \in Act$; $\mathbf{0}$ represents the process that performs no action; for every action in Act, there is a prefix operator; and $+$ is a choice operator.

The operational semantics for BCCSP terms is defined in Figure 2. As usual, we write $p \xrightarrow{a}$ if there exists a process q such that $p \xrightarrow{a} q$, and $p \stackrel{\alpha}{\Longrightarrow} q$ if $\alpha = a_1 \dots a_n$ and $p \xrightarrow{a_1} p' \xrightarrow{a_2} \dots \xrightarrow{a_n} q$. The initial offer of a process is the set $I(p) = \{a \mid a \in Act \text{ and } p \xrightarrow{a}\}$. This is a simple, but quite important observation function that plays a central role in the definition of the most popular semantics in the linear time-branching time (ltbt) spectrum. We will also denote by I the relation expressing the fact that two processes have the same initial offers: $pIq \Leftrightarrow I(p) = I(q)$.

One way to capture semantics is by means of the equivalence relation induced by it: given a formal semantics $[\![\cdot]\!]_Z$, we say that processes p and q are equivalent iff they have the same semantics, that is, $p \equiv_Z q \Leftrightarrow [\![p]\!]_Z = [\![q]\!]_Z$. These semantics can be defined by means of adequate observational scenarios, or by logical characterizations that induce natural preorders $\sqsubseteq_Z$ whose kernels are the semantic equivalences. We refer to [58] for the original definition and usual notation for all the semantics in the ltbt spectrum that will be discussed throughout the paper.

To properly express equations or inequations within the process language, we introduce variables from any adequate set V, and consider the extended set BCCCSP(Act, V) of terms including variables in V.

Some of the semantics in the spectrum are concrete examples of the general notion of constrained simulation semantics that can be defined in a parameterized way.

Definition 2.2. Given a relation N over BCCSP processes, a relation S_N is an N-*constrained simulation* if pS_Nq implies:

- for every $a \in Act$, if $p \xrightarrow{a} p'$ then there exists some q' such that $q \xrightarrow{a} q'$ and $p'S_Nq'$, and
- pNq.

We say that process p is N-simulated by process q, or that q N-simulates p, written $p \sqsubseteq_{NS} q$, whenever there exists an N-constrained simulation S_N such that pS_Nq.

$$ap \xrightarrow{a} p \qquad \frac{p \xrightarrow{a} p'}{p+q \xrightarrow{a} p'} \qquad \frac{q \xrightarrow{a} q'}{p+q \xrightarrow{a} q'}$$

Figure 2: Operational semantics for BCCSP terms.

$$\begin{array}{llll} (B_1) & x+y \simeq y+x & (B_3) & x+x \simeq x \\ (B_2) & (x+y)+z \simeq x+(y+z) & (B_4) & x+\mathbf{0} \simeq x \end{array}$$

Figure 3: The axiomatization for the (strong) bisimulation equivalence.

We have already studied the constrained simulation semantics in detail in [24], stressing their general properties. In particular, the following constraints were considered:

- the universal relation U relating all processes, which gives rise to the simulation semantics;
- the relation C, which holds for processes p and q when both, or none, are isomorphic to $\mathbf{0}$, and that gives rise to the complete simulation semantics;
- the relation I relating processes with the same initial offer, which is the constraint for ready simulation;
- the relation T, that holds for processes having the same set of traces and gives rise to the trace simulation semantics;
- the relation S, the inverse of the simulation equivalence relation, whose associated constrained simulation is the 2-nested simulation.

Throughout this paper there appear different order relations. We use $\sqsubseteq$ to denote semantic preorders and, for the sake of simplicity, we use the symbol $\sqsupseteq$ to represent the preorder relation $\sqsubseteq^{-1}$. With $\equiv$ we denote the induced equivalence (that is, $\sqsubseteq \cap \sqsupseteq$). To refer to a specific preorder we shall append the initials of its name as subscripts to the symbol $\sqsubseteq$ ($\sqsubseteq_{RS}$ for ready simulation, $\sqsubseteq_F$ for failures, and so on). A similar convention applies to the kernels of the preorders ($\equiv_{RS}$, $\equiv_F$, ...) and to the bisimulation equivalence $\equiv_B$.

An *inequation* (respectively, an *equation*) over the language BCCSP is a formula of the form $t \preceq u$ (respectively, $t \simeq u$), where $t, u \in \mathrm{BCCSP}(Act, V)$. An *(in)equational axiom system* is a set of (in)equations over the language BCCSP. An equation $t \simeq u$ is derivable from an equational axiom system E, written $E \vdash t \simeq u$, if it can be proven from the axioms in E using the rules of equational logic (viz. reflexivity, symmetry, transitivity, substitution and closure under BCCSP contexts):

$$t \simeq t \quad \frac{t \simeq u}{u \simeq t} \quad \frac{t \simeq u \quad u \simeq v}{t \simeq v} \quad \frac{t \simeq u}{\sigma(t) \simeq \sigma(u)} \quad \frac{t \simeq u}{at \simeq au} \quad \frac{t \simeq u \quad t' \simeq u'}{t+t' \simeq u+u'}$$

where substitutions σ are defined and applied as usual.

For the derivation of an inequation $t \preceq u$ from an inequational axiom system E, the rule for symmetry—that is, the second rule above—is omitted. We write $E \vdash t \preceq u$ if the inequation $t \preceq u$ can be derived from E.

It is well-known that, without loss of generality, one may assume that substitutions happen first in (in)equational proofs, i.e., that the fourth rule may only be used when its premise is one of the (in)equations in E. Moreover, by postulating that for each equation in E its symmetric counterpart is also present in E, one may assume that applications of symmetry happen first in equational proofs, i.e., that the second rule is never used. In the

remainder of this paper, we shall always tacitly assume that equational axiom systems are closed with respect to symmetry. Note that, with this assumption, there is no difference between the rules of inference of equational and inequational logic. In what follows, we shall consider an equation $t \simeq u$ as a shorthand for the pair of inequations $t \preceq u$ and $u \preceq t$.

An inequation $t \preceq u$ is *sound* with respect to a given preorder relation $\sqsubseteq$, if $t \sqsubseteq u$ holds true. An (in)equational axiom system E is sound with respect to $\sqsubseteq$ if so is each (in)equation in E. An (in)equational axiomatization is called ground-complete if it can prove all the valid (in)equivalences relating terms with no occurrences of variables. As in [58], we abbreviate ground-completeness for completeness because this is the only kind we use along the paper.

Bisimilarity, the strongest of the semantics in the spectrum, can be axiomatized by means of the four simple axioms in Figure 3. These axioms state that the choice operator is commutative, associative and idempotent, having the empty process as identity element. These axioms also justify the use of the notation $\sum_a \sum_i ap_a^i$ for processes, where the commutativity and associativity of the choice operator is used to group together the summands whose initial action is a. We will also write $p|_a$ for the (sub)process we get by projecting all the a-summands of p; that is, if $p = \sum_a \sum_i ap_a^i$, then $p|_a = \sum_i ap_a^i$.

Besides the semantics in the spectrum, we are interested in a general study that can be applied to any "reasonable" semantics coarser than bisimilarity. Since we will use preorders to characterize these semantics we introduce the following definitions that state the desired properties of those reasonable preorders.

Definition 2.3. A preorder relation $\sqsubseteq$ over processes is a *behavior preorder* if

- it is weaker than bisimilarity, i.e., $p \equiv_B q \Rightarrow p \sqsubseteq q$, and
- it is a precongruence with respect to the prefix and choice operators, i.e., if $p \sqsubseteq q$ then $ap \sqsubseteq aq$ and $p + r \sqsubseteq q + r$.

If $\sqsubseteq$ is actually an equivalence, it is said to be a *behavior equivalence*.

Another way of presenting a semantics is by means of a logical characterization. The Hennessy-Milner logic [31], characterizing the bisimulation semantics is the most popular one.

Definition 2.4 (Hennessy-Milner logic, HML). The set $\mathcal{L}_{HM}$ of Hennessy-Milner logical formulas is defined by: if $\varphi, \varphi_i \in \mathcal{L}_{HM}$ for all $i \in I$ and $a \in Act$, then $\bigwedge_{i\in I} \varphi_i$, $a\varphi$, $\neg\varphi \in \mathcal{L}_{HM}$.

The satisfaction relation $\models$ is defined by:

- $p \models a\varphi$ if there exists q such that $p \xrightarrow{a} q$ and $q \models \varphi$;
- $p \models \bigwedge_{i\in I} \varphi_i$ if for all $i \in I : p \models \varphi_i$.
- $p \models \neg\varphi$ if $p \not\models \varphi$.

Note that $\bigwedge_{i\in\emptyset} \varphi_i \in \mathcal{L}_{HM}$, and we have $p \models \bigwedge_{i\in\emptyset} \varphi_i$ for all p. Therefore, in the following we will consider that $\top \in \mathcal{L}_{HM}$, where $\top$ is syntactic sugar for $\bigwedge_{i\in\emptyset} \varphi_i$. The finite version of this logic, $\mathcal{L}_{HM}^f$, uses binary conjunction $\wedge$ instead of the general conjunction $\bigwedge_{i\in I}$. It is well-known that $\mathcal{L}_{HM}^f$ characterizes the bisimulation semantics between image-finite processes, that are those that do not allow infinite branching for any action $a \in Act$ at any state. Van Glabbeek uses $\mathcal{L}_B$ to refer to $\mathcal{L}_{HM}$ in [58].

3. Equational semantics

On Tables 1 and 2 appear the axiomatic characterizations for the preorders and equivalences in van Glabbeek's spectrum [58]. For each column, the set of axioms marked with "+" are sound and complete with respect to the preorder or equivalence in the head of that column; axioms marked with "v" are valid but not needed. When studying these tables there are several questions that naturally arise: for every semantics, is there any connection between the axioms defining the preorder and those for the equivalence? Can the axiomatizations of some of these semantics be jointly tackled?

In this section we will develop new axiomatizations for all the semantics in the ltbt spectrum that offer a clear answer to the previous questions: even if there was not a systematic procedure that led to produce the axiomatizations of those tables, we can obtain equivalent axiomatizations that do follow a given procedure.

These new axiomatizations are obtained after noticing that every process semantics can be understood as the product of two "design decisions", decisions that define what we have called the "dynamic" and the "static" basis of the semantics. We will show that, besides B_1–B_4, we only need a generic simulation axiom (NS)—Proposition 3.1—which characterizes the family of constrained simulation semantics, to axiomatize the whole class of pure branching semantics. Moreover, to characterize the linear time semantics, we only need to add to the corresponding simulation axiom the adequate instantiation of a generic axiom (ND)—see page 14—for reducing the observability of non-determinism in processes, by means of which we introduce the additional identifications induced by each of the linear semantics.

Also the axiomatizations between orders and equivalences are closely related; in fact, in the case of the linear semantics we could just use an equivalence ($ND_{\equiv}$) axiom, leaving the order or equivalence aspect to be determined by the use of the order or equivalence axiom of the corresponding branching semantics, see Figure 5.

In order to justify the form of our axiomatizations without leaving the axiomatic framework, in this section we prove our results with separate and ad-hoc proofs for each semantics just comparing the new characterizations with those previously known. This allows us to quickly get the taste of the underlying relations of the process semantics. Once the unified observational characterization of semantics is presented in Section 4, we will provide generic proofs for these results in Section 5 that show the suitability of the new axiomatizations with respect to the observational characterizations of the semantics.

3.1. **A new axiomatization of the most popular semantics.**

We start our study with a very representative and well-known group of semantics in the spectrum, each of which has been developed and used in important work in the area: ready simulation [38, 13], failures [14, 32], readiness [43], ready trace [9] and failure traces [44].

3.1.1. *Semantic preorders.*

As already hinted above, the dynamic part of the semantics is inherited from a simulation preorder. As stated in our Introduction, bisimilarity can be axiomatized by the set of axioms $B1 - B4$. All the other semantics in the spectrum are coarser than it, and therefore also satisfy these axioms. But due to the fact that bisimulations define equivalence relations and not just preorders, we cannot base on them the characterization of any other interesting semantics. But, plain simulations are somehow defined as half-bisimulations, and can indeed be used as support for the characterizations of

	B	RS	PW	RT	FT	R	F	CS	CT	S	T
$(x+y)+z \simeq x+(y+z)$	+	+	+	+	+	+	+	+	+	+	+
$x+y \simeq y+x$	+	+	+	+	+	+	+	+	+	+	+
$x+0 \simeq x$	+	+	+	+	+	+	+	+	+	+	+
$x+x \simeq x$	+	+	+	+	+	+	+	+	+	+	+
$ax \preceq ax+ay$		+	+	+	+	+	+	v	v	v	v
$a(bx+by+z) \simeq a(bx+z)+a(by+z)$			+	v	v	v	v		v		v
$I(x)=I(y) \Rightarrow ax+ay \simeq a(x+y)$				+	v	v	v		v		v
$ax+ay \succeq a(x+y)$					+		v		v		v
$a(bx+u)+a(by+v) \succeq a(bx+by+u)$						+	v		v		v
$ax+a(y+z) \succeq a(x+y)$							+		v		v
$ax \preceq ax+y$								+	+	v	v
$a(bx+u)+a(cy+v) \simeq a(bx+cy+u+v)$									+		v
$x \preceq x+y$										+	+
$ax+ay \simeq a(x+y)$											+

Table 1: Axiomatization for the preorders in the linear time-branching time spectrum.

some interesting semantics, such as trace semantics. Nevertheless, plain similarity becomes too weak, and some other finer class of simulations is needed to support the characterization of the interesting semantics listed above. Next we recall the axiomatizations of plain, ready and general constrained similarity.

Proposition 3.1 ([58, 24])**.**

(1) *Plain similarity can be axiomatically defined by means of the axiom* (S) $x \preceq x+y$*, together with the axioms* B_1*–*B_4 *that define bisimilarity.*
(2) *Ready similarity can be axiomatically defined by means of the conditional axiom* (RS) $xIy \Rightarrow x \preceq x+y$*, together with* B_1*–*B_4*. It can also be axiomatized by means of the axiom scheme* $ax \preceq ax+ay$*, where* a *represents any arbitrary action.*
(3) *Whenever* N *is a behavior preorder,* N*-similarity can be axiomatically defined by means of the conditional axiom* (NS) $N(x,y) \Rightarrow x \preceq x+y$*, together with* B_1*–*B_4*.*

Let us now consider the diamond of semantics coarser than ready similarity in the ltbt spectrum. It consists of the failures, readiness, failure trace, and ready trace semantics. None of them is a simulation semantics, so their classic axiomatizations (see Table 1) contain an additional axiom:

Failures:	(F)	$a(x+y) \preceq ax+a(y+w)$
Readiness:	(R)	$a(bx+by+u) \preceq a(bx+u)+a(by+v)$
failure trace:	(FT)	$a(x+y) \preceq ax+ay$
ready trace:	(RT)	$I(x)=I(y) \Rightarrow ax+ay \simeq a(x+y)$

Since we are interested in capturing the reduction of observability of non-determinism, our first candidate for a general axiom covering all cases was (FT), which captures the fact that by delaying the choices we get "smaller" processes. However, since this axiom characterizes the failure trace semantics and this is finer than failure semantics, a more general axiom is needed: axiom (F) became our next proposal because failure semantics is the coarsest of the four semantics. More precisely, we expected to achieve the axiomatization of the four semantics in the diamond by adding the adequate instance of the generic

constrained conditional axiom

$$(ND) \quad M(x,y,w) \Rightarrow a(x+y) \preceq ax + a(y+w)\,.$$

This seemed reasonable since the other semantics in the group are finer than failures and by adding a constraint to (F) we certainly obtain a more restricted axiom that produces a finer preorder. The conjecture turned out to be correct and we found that the semantics in the diamond can be characterized by the following instances:

$$\begin{array}{ll}
(ND^F) & M_F(x,y,w) \iff \text{true} \\
(ND^R) & M_R(x,y,w) \iff I(x) \supseteq I(y) \\
(ND^{FT}) & M_{FT}(x,y,w) \iff I(w) \subseteq I(y) \\
(ND^{RT}) & M_{RT}(x,y,w) \iff I(x) = I(y) \text{ and } I(w) \subseteq I(y)
\end{array}$$

Since M_F is the universal relation containing all triples of processes, the corresponding instance of the conditional axiom (ND) is clearly equivalent to (F), and thus adding it to the set $\{B_1\text{–}B_4, (RS)\}$ we obtain a ground-complete axiomatization of $\sqsubseteq_F$. Let us now prove that the remaining three semantics are also axiomatized by the corresponding instances of the axiom (ND) together with (RS).

Proposition 3.2.

(1) *The readiness preorder $\sqsubseteq_R$ is axiomatized by $\{B_1\text{–}B_4, (RS), (ND^R)\}$.*
(2) *The failure trace preorder $\sqsubseteq_{FT}$ is axiomatized by $\{B_1\text{–}B_4, (RS), (ND^{FT})\}$.*
(3) *The ready trace preorder is axiomatized by the set $\{B_1\text{–}B_4, (RS), (ND^{RT})\}$.*

Proof.

(1) Let us show that the set $\{B_1\text{–}B_4, (RS), (ND^R)\}$ is logically equivalent to $\{B_1\text{–}B_4, (RS), (R)\}$. By taking $x = bx' + u$, $y = by'$, and $w = v$ we have that (ND^R) implies (R). In the other direction, let x and y be arbitrary closed BCCSP terms with $I(y) \subseteq I(x)$: we will prove, by structural induction on y, that $\{B_1\text{–}B_4, (RS), (R)\} \vdash a(x+y) \preceq ax + a(y+w)$, for any term w.
   - For $y = \mathbf{0}$, we have $a(x+y) \simeq ax \preceq ax + a(y+w)$, by application of (RS).
   - For $y = by' + y''$, it must be $x = bx' + x''$ and taking $v = y'' + w$ in (R) we obtain $a(x+y) = a(bx' + by' + x'' + y'') \preceq a(x + y'') + a(y+w)$. Then we have $I(y'') \subseteq I(x)$ and we can apply the induction hypothesis to get $\{B_1\text{–}B_4, (RS), (R)\} \vdash a(x+y) \preceq ax + a(y+w)$.

(2) Let us show that the sets $\{B_1\text{–}B_4, (RS), (ND^{FT})\}$ and $\{B_1\text{–}B_4, (RS), (FT)\}$ are logically equivalent. The implication from left to right follows by taking $w = \mathbf{0}$. In the other direction, let w and y with $I(w) \subseteq I(y)$, so that $a(x+y) \preceq ax + ay$ using (FT) and, since $I(y) = I(y+w)$, we have $y \preceq y + w$ using (RS): hence we conclude, $a(x+y) \preceq ax + a(y+w)$.

(3) Let us show that the set $\{B_1\text{–}B_4, (RS), (ND^{RT})\}$ is logically equivalent to $\{B_1\text{–}B_4, (RS), (RT)\}$. We first note that $\{B_1\text{–}B_4, (RS), (RT)\}$ is equivalent to $\{B_1\text{–}B_4, (RS), (RT_\succeq)\}$, where $(RT_\succeq)$ is the axiom $M_{RT}(x,y,w) \Rightarrow ax + ay \succeq a(x+y)$. This follows from the fact that, whenever $I(x) = I(y)$, we can use (RS) to get $x \preceq x + y$ and $y \preceq x + y$, and then $ax + ay \preceq a(x+y)$. Now, the implication from left to right follows by taking $w = \mathbf{0}$. From right to left, as above, whenever $I(w) \subseteq I(y)$ we have $y \preceq y + w$ and then, if $I(x) = I(y)$ we have $a(x+y) \preceq ax + ay$, and therefore $a(x+y) \preceq ax + a(y+z)$.

□

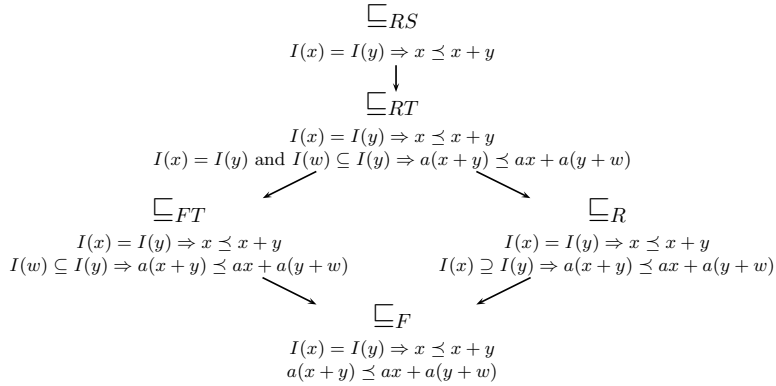


Figure 4: Inclusion relation for the ready simulation preorder and its associated linear semantics.

Figure 4 shows the already known relations between the semantics of the spectrum in the ready simulation layer. However, we want to stress the fact that once the new axiomatizations are proved to be correct, those relations became obvious since the four constraints defined above trivially satisfy $M_{RT}(x, y, w) \Rightarrow M_{FT}(x, y, w) \wedge M_R(x, y, w)$ and $M_{FT}(x, y, w) \vee M_R(x, y, w) \Rightarrow M_F(x, y, w)$. It is even more important that the tight relations and the subtle differences between these semantics clearly stand out by just looking at their axiomatizations.

Certainly, if we compare our new axiomatizations and those in Table 1, the use of conditions in our axioms could be on the grounds that complex conditions could be used to hide the complexity of the semantics. However, the conditions that we have introduced for the alternative axiomatizations of the semantics in the spectrum are very simple. In any case, our main interest was to obtain a uniform presentation of the axiomatizations that could be used to simplify their generic algebraic study.

Corollary 3.3.

(1) $\sqsubseteq_{FT}$ *is axiomatized by the set* $\{B_1\text{–}B_4, (RS), (ND_0^{FT})\}$, *where* (ND_0^{FT}) *is the instance of* (ND^{FT}) *where* w *is* $\mathbf{0}$.

(2) $\sqsubseteq_{RT}$ *is axiomatized by* $\{B_1\text{–}B_4, (RS), (ND_0^{RT})\}$, *where* (ND_0^{RT}) *is the instance of* (ND^{RT}) *where* w *is* $\mathbf{0}$.

Proof. Note that for the proof of Proposition 3.2 only the case $w = \mathbf{0}$ is needed. □

Even if the simplifications above are possible, we prefer to maintain the general forms of the axioms (ND^{FT}) and (ND^{RT}) to keep all axiomatizations as similar as possible, which will come in handy when proving general properties of the semantics.

Corollary 3.4.

(1) $\sqsubseteq_F$ *can be axiomatized by the axioms* $\{B_1\text{–}B_4, (ND^F)\}$.

(2) $\sqsubseteq_R$ *can be axiomatized by the axioms* $\{B_1\text{–}B_4, (ND^R)\}$.

	B	RS	PW	RT	FT	R	F	CS	CT	S	T
$(x+y)+z \simeq x+(y+z)$	+	+	+	+	+	+	+	+	+	+	+
$x+y \simeq y+x$	+	+	+	+	+	+	+	+	+	+	+
$x+0 \simeq x$	+	+	+	+	+	+	+	+	+	+	+
$x+x \simeq x$	+	+	+	+	+	+	+	+	+	+	+
$I(x)=I(y) \Rightarrow a(x+y) \simeq a(x+y)+ay$		+	v	v	v	v	v	v	v	v	v
$a(bx+by+z) \simeq a(bx+z)+a(by+z)$			+	v	v	v	v		v		v
$I(x)=I(y) \Rightarrow ax+ay \simeq a(x+y)$				+	+	v	v		v		v
$ax+ay \simeq ax+ay+a(x+y)$					+		v		v		v
$a(bx+u)+a(by+v) \simeq a(bx+by+u)+a(by+v)$						+	+		v		v
$ax+a(y+z) \simeq ax+a(x+y)+a(y+z)$							+		v		v
$a(x+by+z) \simeq a(x+by+z)+a(by+z)$								+	v	v	v
$a(bx+u)+a(cy+v) \simeq a(bx+cy+u+v)$									+		v
$a(x+y) \simeq a(x+y)+ay$										+	v
$ax+ay \simeq a(x+y)$											+

Table 2: Axiomatization for the equivalences in the linear time-branching time spectrum.

Proof. Note that (ND^F) implies (RS) and therefore (ND^R) implies (RS), by taking $y = \mathbf{0}$ and $w = y$. □

3.1.2. *Equivalences and their preorders.* Let us now study the equivalences and first of all note that the axiom (ND) controlling the reduction of non-determinism has been presented as an inequational axiom. Certainly, it cannot simply be replaced by the corresponding equation since, in general, it is not true that $ax + ay \simeq a(x+y)$. However, the two dimensions corresponding to (RS) and (ND^Z) that control the "growth" of a process with respect to a preorder $\preceq$ are not orthogonal; for example, $a(x+y) \preceq a(x+y)+ax$ can be derived either by an application of (ND^{FT}) or by one of (RS). As a consequence of the relation between these two axioms, once (RS) is assumed then the inequational axiom (ND) can be substituted by its (stronger) equational form

$$(ND_{\equiv}) \quad M(x,y,w) \Rightarrow ax+a(y+w)+a(x+y) \simeq ax+a(y+w)\,.$$

As above, we write $(ND^Z_{\equiv})$ for the concrete instances of this axiom for $Z \in \{F, R, FT, RT\}$.

Proposition 3.5.

(1) *The set* $\{B_1\text{–}B_4, (RS), (ND)\}$ *is logically equivalent to* $\{B_1\text{–}B_4, (RS), (ND_+)\}$*, where* (ND_+) *is the axiom*

$$M(x,y,w) \Rightarrow ax+a(y+w)+a(x+y) \preceq ax+a(y+w)\,.$$

(2) $\{B_1\text{–}B_4, (RS), (ND_+)\}$ *is logically equivalent to* $\{B_1\text{–}B_4, (RS), (ND_{\equiv})\}$.

Proof.

(1) We only need to prove the implication from right to left, since the other follows from $\preceq$ being a precongruence. For that, from (RS) we get $a(x+y) \preceq a(x+y)+ax+a(y+w)$ whence, using (ND_+), $a(x+y) \preceq ax+a(y+w)$.

(2) We only need to prove that, if $M(x, y, w)$, then

$$\{B_1\text{–}B_4, (RS), (ND_+)\} \vdash ax + a(y + w) \preceq ax + a(y + w) + a(x + y)\,,$$

which follows from (RS).

□

This result can be interpreted as saying that the only way to "enlarge" a process is by extending its possible behaviors by means of the "dynamic" simulation axioms; the static rules, (ND) and its variants, instead generate new identifications among processes.

Actually, any complete axiomatization of a preorder that contains the axiom (RS) can be turned into an equivalent axiomatization by replacing every inequality $u \preceq v$ by $u+v \simeq v$.

Proposition 3.6. *Let $Q = \{B_1\text{–}B_4, (RS)\} \cup Q'$ be an axiomatization of an order $\sqsubseteq$ such that $\sqsubseteq \subseteq I$. Then, the equational variant of Q, $Q^{=} = \{B_1\text{–}B_4, (RS)\} \cup \{M \Rightarrow u + v \simeq v \mid M \Rightarrow u \preceq v \in Q'\}$ is also an axiomatization of $\sqsubseteq$.*

Proof. Analogous to the particular case considered in Proposition 3.5 above. For the sake of clarity we have preferred to present the particular case before, because it is easily stated and it corresponds to the most important instance of the general result. □

Finally, to conclude this section we gather in Table 2 axiomatic characterizations for the semantic equivalences that are an alternative to the classic axioms appearing in [58].

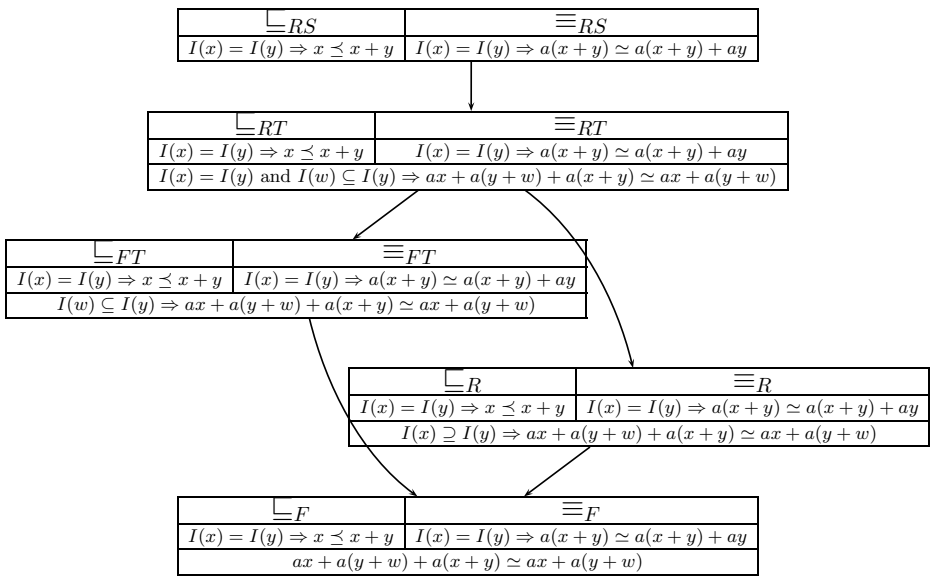


Figure 5: Axioms for the ready simulation layer of semantics.

Following the same ideas that we have already discussed for the preorders, a key point is to find the equations that characterize the simulation equivalence that governs each layer.

As showed in [24], there is a generic axiom that we can use:

$$(NS_{\equiv}) \quad N(x,y) \Rightarrow a(x+y) \simeq a(x+y) + ay.$$

We consider the instantiated equation that characterizes the ready simulation equivalence:

$$(RS_{\equiv}) \quad I(x) = I(y) \Rightarrow a(x+y) \simeq a(x+y) + ay,$$

and the rest of the characterization follows by using the equation $(ND_{\equiv})$ presented above.

Proposition 3.7.

(1) *The failure equivalence* $\equiv_F$ *is axiomatized by* $\{B_1\text{–}B_4, (RS_{\equiv}), (ND^F_{\equiv})\}$.
(2) *The readiness equivalence* $\equiv_R$ *is axiomatized by* $\{B_1\text{–}B_4, (RS_{\equiv}), (ND^R_{\equiv})\}$.
(3) *The failure trace equivalence* $\equiv_{FT}$ *is axiomatized by* $\{B_1\text{–}B_4, (RS_{\equiv}), (ND^{FT}_{\equiv})\}$.
(4) *The ready trace equivalence* $\equiv_{RT}$ *is axiomatized by the set* $\{B_1\text{–}B_4, (RS_{\equiv}), (ND^{RT}_{\equiv})\}$.

Proof. To prove these results we can compare the new and old axiomatizations similarly as we did in the proof of Proposition 3.5 or alternatively make use of the "ready to preorder" algorithm thoroughly studied in [4, 18, 21]. □

The results in this section clarify the entanglement between axiomatizations for preorders and equivalences. For example: for the ready simulation and its associated linear semantics, we just need three axioms (RS), $(RS_{\equiv})$ and $(ND_{\equiv})$—conveniently instantiated—to characterize the 10 relations (orders and equivalences) involved, as summarized in Figure 5.

3.2. **The coarsest semantics in the spectrum.** The results in Section 3.1 show the relations between the ready simulation and the linear semantics naturally associated to it. The same phenomenon occurs for other simulations. In this section we focus on the bottom part of the spectrum where lie the simulation semantics coarser than ready simulation: plain and complete simulation, and the semantics coarser than these. For the simulation semantics we obtain the corresponding axiomatizations simply by considering the universal constraint for the case of plain simulations and the complete constraint for complete simulations:

Simulation	$U(x,y) \iff \mathit{true}$
Complete simulations	$C(x,y) \iff (x = \mathbf{0} \text{ iff } y = \mathbf{0})$

Trace and completed trace semantics can be defined by simply adding our axiom (ND^F) to the appropriate instance of

$$(NS) \quad N(x,y) \Rightarrow x \preceq x + y.$$

Proposition 3.8.

(1) $\sqsubseteq_T$ *is axiomatized by the axioms*[3] $\{B_1\text{–}B_4, (S), (ND^F)\}$.
(2) $\sqsubseteq_{CT}$ *is axiomatized by the axioms* $\{B_1\text{–}B_4, (CS), (ND^F)\}$, *where* (CS) *is the instantiation of* (NS) *taking* $C(x,y)$ *as* $N(x,y)$.

Proof.

[3]Note that (S) is equivalent to (US), the instantation of (NS) with U as N.

(1) The classic axiomatization of trace semantics is given by $\{B_1$–$B_4, (S), (T)\}$, where (T) is the axiom $ax + ay \simeq a(x + y)$. Note that $\{B_1$–$B_4, (S), (T)\}$ is logically equivalent to $\{B_1$–$B_4, (S), (T_{\sqsubseteq})\}$, where $(T_{\sqsubseteq})$ is the axiom $a(x + y) \preceq ax + ay$, because (S) can be used to obtain $ax \preceq a(x + y)$ and $ay \preceq a(x + y)$. And it is immediate that (ND^F) implies $(T_{\sqsubseteq})$. Also, $\{(S), (T_{\sqsubseteq})\} \vdash a(x + y) \preceq ax + a(y + w)$, since $a(x + y) \preceq ax + ay$ by $(T_{\sqsubseteq})$ and $ax + ay \preceq ax + a(y + w)$ by (S).
(2) Analogous to the previous case once we realize that the classic axiom for completed trace, (CT) $a(bx+u)+a(cy+v) \simeq a(bx+cy+u+v)$, is equivalent to the conditional axiom $C(x, y) \Rightarrow ax+ay \simeq a(x+y)$. This follows because $bx+u$ and $cy+v$ are two independent patterns describing non-null processes and when the condition is instantiated with x and y equal to $\mathbf{0}$ the identity is trivial: $a\mathbf{0} + a\mathbf{0} \simeq a\mathbf{0}$. □

By an argument analogous to that in Proposition 3.5, we can obtain for $\sqsubseteq_T$ the axiomatization $\{B_1$–$B_4, (S), (ND^F_{\equiv})\}$. Note that although $(ND^F_{\equiv})$ is an equation, this axiomatization is not the classic one; obviously, (T) $ax + ay = a(x + y)$ implies $(ND^F_{\equiv})$ but the converse is false.

It is easy to check that in the case of trace semantics, the particular instance (ND_0) of the axiom (ND) with w equal to $\mathbf{0}$ is powerful enough to generate the trace preorder. This was certainly not the case when we were under ready simulation, where (ND_0) just generates the failure trace preorder instead of the coarser failures preorder.

It is also interesting to note that for the trace semantics the symmetric version of (ND),

$$(ND_{vw}) \quad a(x + y) \preceq a(x + v) + a(y + w),$$

is also valid, so we can take both $\{B_1$–$B_4, (S), (ND_{vw})\}$ and $\{B_1$–$B_4, (S), (ND^{\equiv}_{vw})\}$, where

$$(ND^{\equiv}_{vw}) \quad a(x + v) + a(y + w) + a(x + y) \simeq a(x + v) + a(y + w),$$

as alternative axiomatizations of the trace preorder.

Should we expect another diamond of "reasonable" semantics under plain simulation in the spectrum? Were that to be the case, why have we only found the trace semantics?

In order to answer these questions, note that the diamond of semantics under ready simulation was completely governed by the function I, which appears in the constraints of the different instantiations of the axiom (ND). For plain simulations, however, the trivially true predicate $U(x, y)$ corresponds to the observation function that can see nothing. As a consequence, if we substitute U for I in each of the four constraints of the diamond they all collapse into a single one: trace semantics. Nevertheless, an alternative path can be explored to obtain new semantics: let us keep the different axioms (ND^Z) the way they stand and simply replace (RS) by (S). Then we obtain the following results:

Proposition 3.9. *$\{B_1$–$B_4, (S), (ND^{FT})\}$ is another axiomatization of trace semantics. Hence, under (S) the failures and the failure trace axioms generate the same preorder, namely the trace preorder.*

Proof. $\{B_1$–$B_4, (S), (ND_0)\}$ is a complete axiomatization of trace preorder, and (ND_0) is a particular case of (ND^{FT}). □

The axioms corresponding to readiness and ready trace, however, give rise to two new semantics that we shall name *extended ready* and *extended ready trace* semantics. They are defined by the order obtained by inclusion of the offers of the processes, either just at the end of a trace, or after each action within it: in order to have $p \sqsubseteq_{ER} q$, for each $p \stackrel{\alpha}{\Longrightarrow} p'$

with $I(p') = R$ we need some $q \stackrel{\alpha}{\Longrightarrow} q'$ with $I(q') \supseteq R$; the extended ready trace preorder $\sqsubseteq_{ERT}$ is defined analogously, but using ready traces.

Proposition 3.10.
(1) *The set* $\{B_1\text{–}B_4, (S), (ND^R)\}$ *is an axiomatization of* $\sqsubseteq_{ER}$.
(2) *The set* $\{B_1\text{–}B_4, (S), (ND^{RT})\}$ *is an axiomatization of* $\sqsubseteq_{ERT}$.

Let us now consider the versions of the axioms (ND^R), (ND^{FT}), (ND^{RT}) where the constraint I has been replaced by the completeness condition C defined by $C(x) \Longleftrightarrow x = \mathbf{0}$:

$$\begin{array}{ll}
(C\text{-}ND^R) & M_{CR}(x,y,w) \iff (C(x) \text{ implies } C(y)) \\
(C\text{-}ND^{FT}) & M_{CFT}(x,y,w) \iff (C(y) \text{ implies } C(w)) \\
(C\text{-}ND^{RT}) & M_{CRT}(x,y,w) \iff \big((C(x) \text{ iff } C(y)) \text{ and } (C(y) \text{ implies } C(w))\big)
\end{array}$$

Once again, we simply obtain three alternative axiomatizations of the completed trace semantics.

Proposition 3.11. *The following axiomatizations are equivalent:*
(1) $\{B_1\text{–}B_4, (CS),\ (ND^F)\}$.
(2) $\{B_1\text{–}B_4, (CS), (C\text{-}ND^R)\}$.
(3) $\{B_1\text{–}B_4, (CS), (C\text{-}ND^{FT})\}$.
(4) $\{B_1\text{–}B_4,\ (CS), (C\text{-}ND^{RT})\}$.

Proof. Clearly, (1) ⇒ (2) ⇒ (3) ⇒ (4) and therefore it is enough to prove that (4) ⇒ (1). If x and y are not $\mathbf{0}$ we can apply $(C\text{-}ND^{RT})$ to obtain the inequality in (ND^F). If x is $\mathbf{0}$ but y is not, we need to obtain $ay \preceq a\mathbf{0} + a(y+w)$. By (CS) we have $y \preceq y + w$ and then $ay \preceq a(y+w)$; applying (CS) again, $a(y+w) \preceq a(y+w) + a\mathbf{0}$ and thus $ay \preceq a\mathbf{0} + a(y+w)$. If y is $\mathbf{0}$ but x is not, we need to obtain $ax \preceq ax + aw$, which results from an immediate application of (CS). Finally, if both x and y are $\mathbf{0}$, $a\mathbf{0} \preceq a\mathbf{0} + aw$. □

As before, if we consider the original axioms (ND^R), (ND^{FT}), and (ND^{RT}) we obtain, together with an alternative axiomatization of the completed trace semantics, two new semantics.

Proposition 3.12. *The set* $\{B_1\text{–}B_4, (CS), (ND^{FT})\}$ *is logically equivalent to* $\{B_1\text{–}B_4, (CS), (ND^F)\}$. *Hence, under* (CS), *the failures and the failure trace axioms generate the same semantics.*

Proof. It is enough to prove that $(C\text{-}ND^{FT})$ can be derived from $\{B_1\text{–}B_4, (CS),\ (ND^F)\}$.

- If y is $\mathbf{0}$ we then have w equal to $\mathbf{0}$ and can apply (ND^{FT}).
- If y is not $\mathbf{0}$ we can apply (ND_0^{FT}) to obtain $a(x+y) \preceq ax + ay$ and then (CS) to conclude that $a(x+y) \preceq ax + a(y+w)$. □

By contrast, as happened for plain simulations, under (CS) the axioms of the ready semantics generate two slightly different versions of the extended ready and extended ready trace semantics introduced before, that we call *extended complete ready* and *extended complete ready trace* semantics. In order to have $p \sqsubseteq_{ECR} q$, whenever $p \stackrel{\alpha}{\Longrightarrow} p'$ with $I(p') \neq \emptyset$ we require some $q \stackrel{\alpha}{\Longrightarrow} q'$ with $I(q') \supseteq I(p')$, but if $I(p') = \emptyset$ then the corresponding q' also has to satisfy $I(q') = \emptyset$. The extended complete ready trace preorder $\sqsubseteq_{ECRT}$ is defined in an analogous way, starting from the ready traces of the processes.

As we did in Section 3.1.2, we can prove that the axioms that characterize trace and completed trace preorders reflect the fact that the order relation is inherited from simulation

and complete simulation, respectively, and that the role of the static rules is to introduce identifications. As stated in Proposition 3.12 above, the only inequation that we use to axiomatize the trace and completed trace orders is (S), the remaining axioms being equational axioms.

Proposition 3.13.
(1) $\{B_1\text{–}B_4, (S), (ND^F)\}$ *is logically equivalent to* $\{B_1\text{–}B_4, (S), (ND^F_{\equiv})\}$.
(2) $\{B_1\text{–}B_4, (CS), (ND^F)\}$ *is logically equivalent to* $\{B_1\text{–}B_4, (CS), (ND^F_{\equiv})\}$.

A similar discussion could have been carried out for trace and completed trace equivalences, and indeed a very natural axiomatization for these relations can be obtained based on the corresponding instantiation of the $(NS_{\equiv})$ equation:

$$(S_{\equiv}) \quad a(x+y) \simeq a(x+y) + ay$$
$$(CS_{\equiv}) \quad C(x,y) \Rightarrow a(x+y) \simeq a(x+y) + ay\,.$$

Proposition 3.14.
(1) *The trace equivalence* $\equiv_T$ *is axiomatized by* $\{B_1\text{–}B_4, (S_{\equiv}), (ND^F_{\equiv})\}$.
(2) *The completed trace equivalence* $\equiv_{CT}$ *is axiomatized by* $\{B_1\text{–}B_4, (CS_{\equiv}), (ND^F_{\equiv})\}$.

To conclude this section devoted to the unification of the equational characterizations of process semantics, we present in Figure 6 a condensed view of our new spectrum. This presentation exploits in an expressive way the two dimensions of the picture, which in fact reflects a tridimensional structure. On the lefthand side the constrained simulations and bisimulations appear, totally ordered from top to bottom. Each constrained simulation generates a layer of semantics. Here, we have only detailed the layers corresponding to ready simulation and that of plain simulation. As a matter of fact, the latter degenerates to a single point due to the simplicity of the constraint U governing plain simulations. The naturality of the semantics appearing in this part of the spectrum is illustrated by our generic axiomatization, where a single (constrained) simulation axiom governs all the constrained simulation semantics, whereas adding a single axiom we complete the axiomatizations of each of the linear semantics at the righthand side of the picture.

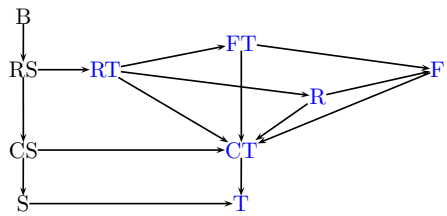


Figure 6: New view of the linear time-branching time spectrum.

4. Observational semantics

Along Section 3 we have presented some views of the axiomatizations for process semantics that highlight the common properties and the subtle differences between them; likewise these

views of the axiomatic characterizations point out the similarities between the preorder and the equivalence of a given semantics.

In this section we focus on the characterizations of process semantics based on observations. Indeed, this idea of determining the semantics by means of observations lies deep inside the foundations of process theory.

> *Our calculus is founded in two central ideas. The first is observation; [...] two systems are indistinguishable if we cannot tell them apart without pulling them apart. We therefore give a formal definition of observation equivalence and investigate its properties.* [41]
>
> *Imagine there is an observer with a notebook who watches the process and writes down the name of each event as it occurs.* [32]

Besides the classical references to Milner and Hoare, this idea of observation pervades the Hennessy's testing methodology [30] and most of the work on linear semantics. Observations, in spite of the variations in different proposals, constitute a denotational space closely related to the classical developments of semantics based on denotations for programming languages [52].

In this section we will show how most of the semantics can be characterized with one of the two main families of observations:

- Branching general observations, Section 4.1, that are essentially labeled trees, that characterize the simulation semantics: simulation, complete simulation, ready simulation, nested simulation, ...
- Linear observations, a simplified case of branching observations, Section 4.2, that characterize the linear semantics: traces, failures, readiness, ready trace, ...

We consider also in Section 4.3 a more exotic kind of observations, deterministic branching observations, which are essentially deterministic trees. Possible worlds semantics is the only semantics appearing in the classical spectrum in this class, although, our general approach will show how this kind of observations define new full families of process semantics.

To develop this observational characterization for process semantics allows us to deepen into the ultimate nature of the similarities and differences between them. Along this section we present a thorough study of the local observation functions that generate the local observations of the states, Figure 10. For the linear case, there is also the possibility of observing this local information in a partial way and this is how for each local observer, in principle, up to four different semantics can be obtained. This fact explains the classic diamond below the ready simulation semantics formed by the failures, failure trace, readiness, and ready trace semantics. Again, the generality of our study makes it exportable to other simulation layers enriching and completing the spectrum of semantics, Figure 11.

Finally, from a methodological point of view, the unification of observational semantics that we present in this section introduces all the technical machinery needed to rewrite the proofs of Section 3 in a generic way, proving that the two unification procedures produce characterizations of the same semantics. We will address this topic in Section 5. Let us now concentrate on the observational semantics.

4.1. **Branching general observations.** In order to characterize the simulation semantics in an extensional way we need local and branching general observations.

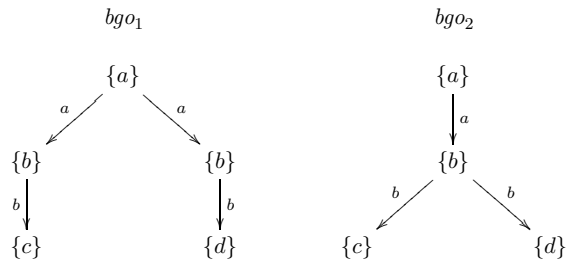


Figure 7: Two branching observations.

Definition 4.1. The sets L_N of *local observations* corresponding to each of the constrained simulations in the spectrum, and $L_N(p)$ of observations associated to a process p, are defined as follows:

- Universal (or Plain) simulation: $L_U = \{\cdot\}$; $L_U(p) = \cdot$.
- Ready simulation: $L_I = \mathcal{P}(Act)$; $L_I(p) = I(p)$.
- Complete simulation: $L_C = Bool$; $L_C(p)$ is *true* if $I(p) = \emptyset$ and *false* otherwise.
- Trace simulation[4]: $L_T = \mathcal{P}(Act^*)$; $L_T(p) = T(p)$, the set of traces of p.
- 2-nested simulation: $L_S = \{[\![p]\!]_S \mid p \in BCCSP\}$; $L_S(p) = [\![p]\!]_S$, where $[\![p]\!]_S$ represents the equivalence class of p with respect to the simulation equivalence.

Definition 4.2.

(1) A *branching general observation* (bgo for short) of a process is a finite, non-empty tree whose arcs are labeled with actions in Act and whose nodes are labeled with local observations from L_N, for N a constraint; the corresponding set BGO_N is recursively defined as:
   - $\langle l, \emptyset\rangle \in BGO_N$ for $l \in L_N$.
   - $\langle l, \{(a_i, bgo_i) \mid i \in 1..n\}\rangle \in BGO_N$ for every $n \in \mathbb{N}$, $a_i \in Act$ and $bgo_i \in BGO_N$.

(2) The set $BGO_N(p)$ of branching general observations of p corresponding to the constraint N is

$$BGO_N(p) = \{\langle L_N(p), S\rangle \mid S \subseteq \{(a, bgo) \mid bgo \in BGO_N(p'), p \xrightarrow{a} p'\}\}.$$

(3) We write $p \leq^b_N q$ if $BGO_N(p) \subseteq BGO_N(q)$.

In Figure 7 some simple examples of bgo's for $N = I$ are shown. We represent bgo_1 as

$$\langle\{a\}, \{(a, \langle\{b\}, \{(b, \langle\{c\}, \emptyset\rangle)\}\rangle), (a, \langle\{b\}, \{(b, \langle\{d\}, \emptyset\rangle)\}\rangle)\}\rangle$$

and bgo_2 as

$$\langle\{a\}, \{(a, \langle\{b\}, \{(b, \langle\{c\}, \emptyset\rangle), (b, \langle\{d\}, \emptyset\rangle)\}\rangle)\}\rangle.$$

We use braces for the set of children of a node, parentheses to represent a branch of the tree as a pair (initial arc, subtree below), and angular brackets to represent each tree as a pair $\langle \text{root}, \text{children}\rangle$.

[4]Trace simulations are the only ones in this list that do not appear in [58]. They can be defined as T-simulations, with $T(x, y) ::= T(x) = T(y)$, and the general theory about constrained simulations in [24] applies to them. In particular, they can be axiomatized as stated in Proposition 3.1(3), page 13.

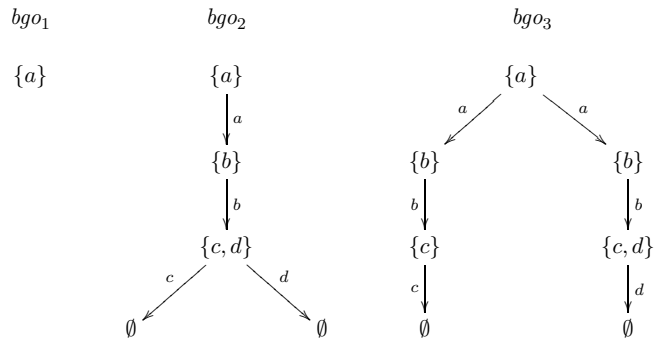


Figure 8: Three branching observations.

Note that the bgo's of a process p described by its transition system can be generated by inductively applying the clauses defining the set $BGO_N(p)$, even when p is infinite. For instance, if $N = I$ and we consider the process $p ::= c.p$ defining a clock, since $\emptyset \subseteq \{(c, bgo) \mid bgo \in BGO_I(p), p \xrightarrow{c} p\}$, it follows that $\langle\{c\}, \emptyset\rangle \in BGO_I(p)$. But now $\{(c, \langle\{c\}, \emptyset\rangle)\} \subseteq \{(c, bgo) \mid bgo \in BGO_I(p), p \xrightarrow{c} p\}$ and therefore $\langle\{c\}, \{(c, \langle\{c\}, \emptyset\rangle)\}\rangle \in BGO_I(p)$, and so on.

It is clear that the bgo's of a process have an operational flavor. The nodes of the observations correspond to its states and the arcs to its transitions; this is why we will be able to define the orders associated to the different simulation semantics simply by set inclusion over the sets of bgo's.

Let us also comment on the fact that in all five cases that we have considered in Definition 4.1, which correspond to the five constrained simulation semantics in the spectrum, the local observation functions L_N define a representation of the equivalence relation N used to define the constrained simulation relations. This means that we have $L_N(p) = L_N(q) \iff pNq$.

Example 4.3. For $N = I$, if $x = b(c + d)$ and $y = bc + bd$, then for $p = a(x + y)$ we have $bgo_k \in BGO_I(p)$ for $k \in \{1, 2, 3\}$, where the bgo's are depicted in Figure 8. It is easy to check that all of them are also branching observations of $q = a(x + y) + ax$. As a matter of fact, we have $BGO_I(p) = BGO_I(q)$. Note that in order to obtain $bgo_3 \in BGO_I(p)$ we need to combine two different observations of the process $x + y$, which is the only p' such that $a(x + y) \xrightarrow{a} p'$.

In contrast, for $p = a(bc + bd)$ and $q = abc + abd$, $BGO_I(q) \not\subseteq BGO_I(p)$, since for the branching observation bgo_1 in Figure 7 we have $bgo_1 \in BGO_I(q)$ and $bgo_1 \notin BGO_I(p)$. And also, we have $BGO_I(p) \not\subseteq BGO_I(q)$, since for bgo_2 as in Figure 7 we have $bgo_2 \in BGO_I(p)$, but $bgo_2 \notin BGO_I(q)$. The key idea is that we can indeed include in a single bgo two separated computations but we cannot "mix" two different ones, even if the labels both in their initial transitions and in the local observations of the reached nodes were the same. This is why $bgo_2 \notin BGO_I(q)$.

The following simple properties will be immediate consequences of Theorem 4.9 below; we use them here to illustrate the expressive power of each kind of bgo.

Definition 4.4. An axiom $t \preceq u$, respectively $t \simeq u$, is satisfied in a model BGO_N if $BGO_N(t') \subseteq BGO_N(u')$, respectively $BGO_N(t') = BGO_N(u')$, for every possible ground instantiation $t' \preceq u'$ or $t' \simeq u'$ of the axiom.

Proposition 4.5.

(1) *The axiom* (S) $x \preceq x + y$ *is satisfied in the model* BGO_U.
(2) *The axiom* $(S_{\equiv})$ $a(x + y) \simeq a(x + y) + ax$ *is satisfied in the model* BGO_U.

Proof.

(1) It is an immediate consequence of the fact that if $p \xrightarrow{a} p'$ then $p + q \xrightarrow{a} p'$, and therefore $\{a \mid p \xrightarrow{a}\} \subseteq \{a \mid p + q \xrightarrow{a}\}$.
(2) Again, it is a simple exercise to check that $BGO_U(p) \subseteq BGO_U(q)$ implies $BGO_U(ap) \subseteq BGO_U(aq)$, and that if $BGO_U(p), BGO_U(q) \subseteq BGO_U(r)$, then $BGO_U(p+q) \subseteq BGO_U(r)$; in combination with (1), this produces the result. □

Proposition 4.6. $BGO_I(p) \subseteq BGO_I(p + q)$ *iff* $I(q) \subseteq I(p)$.

Proof. ($\Leftarrow$) Since $I(p + q) = I(p)$, the root of the bgo's is the same for both processes and obviously $p + q$ has all the observations of p.

($\Rightarrow$) If $I(q) \not\subseteq I(p)$, then $I(p) \neq I(p + q)$ and then no bgo of p is a bgo of $p + q$ because the roots of the observations of both processes are different. □

The fact, that we now prove, that the observational semantics $BGO_N(p)$ can be defined in a compositional way, is an important property that will simplify the proofs of many of their properties.

Theorem 4.7. *Let* L *be a function used as a local observation function and let us also denote by* L *the range of* L*, as done in Definition 4.1. If there exist semantic functions* $+^L : L \times L \to L$ *and* $a^L : L \to L$ *satisfying* $L(ap) = a^L L(p)$ *and* $L(p + q) = L(p) +^L L(q)$*, then:*

- $BGO_N(ap) = \{\langle a^L L(p), \{(a, bgo) \mid bgo \in B\}\rangle \mid B \subseteq BGO_N(p)\}$.
- $BGO_N(p+q) = \{\langle L(p) +^L L(q), S_1 \cup S_2\rangle \mid \langle L(p), S_1\rangle \in BGO_N(p), \langle L(p), S_2\rangle \in BGO_N(q)\}$.

Proof. The first equality is immediate by definition of $BGO_N(ap)$. As for the second, we only need to realize that $p + q \xrightarrow{a} r$ iff $p \xrightarrow{a} r$ or $q \xrightarrow{a} r$: then, the set of children of the root labeled by $L_N(p+q)$ at any bgo $\in BGO_N(ap)$ correspond to the union of the two sets of children that contain some bgo's of processes p_i such as $p \xrightarrow{a} p_i$ (and then $p + q \xrightarrow{a} p_i$) or q_i such that $q \xrightarrow{a} q_i$ (and then $p + q \xrightarrow{a} q_i$). Note that from the equalities above it follows that $BGO_N(p)$ can be computed compositionally. □

In particular, $BGO_N(p)$ is compositional for any of the constraints considered in Definition 4.1.

Proposition 4.8. *For* $N \in \{U, I, C, T, S\}$*,* L_N *can be defined in a compositional way over the terms in BCCSP.*

Proof. The result for U is obvious since it is a degenerate semantics that identifies all processes. By Theorem 4.7 and Theorem 4.9 below we can conclude that the simulation semantics can indeed be denotationally defined. The result for traces is well-known, while I and C can be easily defined denotationally since $I(ap) = \{a\}$ and $I(p + q) = I(p) \cup I(q)$. □

Now we show that bgo's characterize N-simulation semantics in all cases.

Theorem 4.9. *For all $N \in \{U, I, C, T, S\}$ and any two processes p and q, $p \sqsubseteq_{NS} q$ iff $p \leq^b_N q$.*

Proof. ($\Rightarrow$) Let $p = \sum\sum ap^i_a$ and $q = \sum\sum aq^j_a$; if $p \sqsubseteq_{NS} q$, then pNq and therefore $L_N(p) = L_N(q)$. Now we proceed by induction on p. If $p \equiv \mathbf{0}$ the result is trivial. Otherwise, for every $a \in I(p)$ such that $p \xrightarrow{a} p'$ there exists $q \xrightarrow{a} q'$ such that $p' \sqsubseteq_{NS} q'$. By induction hypothesis $BGO_N(p') \subseteq BGO_N(q')$ from where, by the definition of $BGO_N(p)$, it follows that $BGO_N(p) \subseteq BGO_N(q)$.

($\Leftarrow$) Let us show that the relation $R = \{(p, q) \mid BGO_N(p) \subseteq BGO_N(q)\}$ is an N-simulation. If $(p, q) \in R$, then $L_N(p) = L_N(q)$ because $\langle L_N(p), \emptyset\rangle \in BGO_N(q)$ and thus pNq. Now, for each $p \xrightarrow{a} p'$ we have $\{\langle L_N(p), \{(a, bgo)\}\rangle \mid bgo \in BGO_N(p')\} \subseteq BGO(q)$ and therefore there must exist some $q \xrightarrow{a} q'$ such that $BGO_N(p') \subseteq BGO_N(q')$, so that $(p', q') \in R$. □

Note that for this result to hold it is only required that the local observation function L_N satisfies pNq iff $L_N(p) = L_N(q)$. That is, L_N must compute a concrete representative of the equivalence class defined by N and this stresses again the interest of using behavior equivalences N as constraints for the definition of constrained simulations. Let us recall that, in principle, any behavior preorder could be used as such a constraint. For instance, the predicate $I_\supseteq$ defined by $pI_\supseteq q$ iff $I(q) \subseteq I(p)$ could be used to define $I_\supseteq$-simulations (which in fact coincide with I-simulations). But from $I(q) \subseteq I(p)$ we cannot conclude that $L_N(p) = L_N(q)$ and, hence, either a more complicated characterization of $\sqsubseteq_{NS}$ in terms of bgo's or an additional argument to show that $p \sqsubseteq_{I_\subseteq} q$ implies $I(p) \subseteq I(q)$ would be needed. And although this is obvious for a constraint as simple as I, or even T or S, it could be far from trivial for other, more complex constraints: therefore, it is always advisable to consider equivalence behaviors as constraints.

Corollary 4.10. *For any constraint N that is a behavior equivalence, whenever we have as local observation function L_N the quotient function $L_N(p) = [\![p]\!]_N$ or any concrete representation of it satisfying $L_N(p) = L_N(q) \iff pNq$, then $p \sqsubseteq_{NS} q$ iff $BGO_N(p) \subseteq BGO_N(q)$.*

The results above bring forward the fact that despite the resemblance between the bgo's of a process and its computation tree, the possibility of mixing several computations in a single branching observation makes it possible to identify non-bisimilar processes by their sets of branching observations.

4.2. **Linear observations and linear time semantics.** We introduce the linear observations of a process as a particular (degenerate) case of branching observations: those with a linear structure.

Definition 4.11.

(1) The set LGO_N of *linear general observations* (lgo for short) for a local observer L_N is the subset of BGO_N defined as:
   - $\langle l, \emptyset\rangle \in LGO_N$ for each $l \in L_N$.
   - $\langle l, \{(a, lgo)\}\rangle$, whenever $a \in A$ and $lgo \in LGO_N$.

(2) The set of *linear general observations* of a process p with respect to the local observer L_N is $LGO_N(p) = BGO_N(p) \cap LGO_N$.

Since lgo's are linear they can be presented as traces, avoiding the sets of descendants in the bgo's. Therefore, we will consider them as elements of the set $L_N \times (Act \times L_N)^*$.

It is also clear that the set of linear observations can be defined recursively without resorting to branching observations.

Definition 4.12. The set $LGO_N(p)$ of linear general observations of a process p is recursively defined by

$$LGO_N(p) = \{\langle L_N(p)\rangle\} \cup \{\langle L_N(p), a\rangle \circ lgo \mid p \xrightarrow{a} p',\ lgo \in LGO_N(p')\}.$$

We can also compute $LGO_N(p)$ in a compositional way.

Proposition 4.13. *Let L be a local observation function such that there exist semantic functions $+^L : L_N \times L_N \to L_N$ and $a^L : L_N \to L_N$ satisfying $L(ap) = a^L L(p)$ and $L(p+q) = L(p) +^L L(q)$. Then:*

- $LGO_N(ap) = \{\langle a^L L(p)\rangle\} \cup \{\langle a^L L(p), a\rangle \circ LGO_N(p)\}$.
- $LGO_N(p+q) = \{\langle L(p) +^L L(q)\rangle \circ t \mid \langle L(p)\rangle \circ t \in LGO_N(p)$ *or* $\langle L(p)\rangle \circ t \in LGO_N(q)\}$.

Proof. Just like that of Theorem 4.7. □

Obviously, for $N = U$ we have that LGO_U is isomorphic to Act^* and thus $LGO_U(p) = Traces(p)$. By contrast, for $N = I$, $LGO_I(p)$ is the set of ready traces of p, $ReadyTraces(p)$.

Set inclusion of linear observations with respect to a local observer L_N gives us the preorder defining the corresponding semantics.

Definition 4.14. A process p is less than or equal to q with respect to the linear observations generated by L_N, denoted $p \leq^l_N q$, if $LGO_N(p) \subseteq LGO_N(q)$. We will denote the corresponding equivalence by $=^l_N$.

Proposition 4.15. *(1) $\leq^l_U = \sqsubseteq_T$; (2) $\leq^l_I = \sqsubseteq_{RT}$; (3) $\leq^l_C = \sqsubseteq_{CT}$.*

Proof. It is trivial, since $LGO_U(p) = Traces(p)$, $LGO_I(p) = ReadyTraces(p)$, and $LGO_C(p) = \{(false, a_1)\circ \ldots \circ (false, a_n, true), (false, a_1)\circ \ldots \circ (false, a_i, false) \mid a_1 \ldots a_n \in CompleteTraces(p), i < n\}$. □

Proposition 4.16. *For $N \in \{U, C, I, T, S\}$, if $p \sqsubseteq_{NS} q$ then $p \leq^l_N q$, but the converse is false in general.*

Proof. The implication follows from Theorem 4.9 and the fact that lgo's are just a particular case of bgo's. To see that the converse is false in general consider $N = U$; we have $\sqsubseteq_{US} = \sqsubseteq_S$ and $\leq^l_U = \sqsubseteq_T$, and it is well-known that $\sqsubseteq_S \not\subseteq \sqsubseteq_T$ since, for instance, $a(b+c) \not\sqsubseteq_S ab + ac$, but $a(b+c) =_T ab + ac$. □

Therefore, by means of linear observations and set inclusion we can characterize the orders that define some of the semantics in the spectrum which are not simulation semantics. However, there are still some other semantics for which a different way of treating the linear observations is needed. We need to introduce some identifications in the corresponding domain LGO_N to obtain their characterizations.

Definition 4.17. For $\mathcal{T}, \mathcal{T}' \subseteq LGO_I$ we define the orders $\leq^{l\supseteq}_I$, $\leq^{lf}_I$, and $\leq^{lf\supseteq}_I$ by:

- $\mathcal{T} \leq^{l\supseteq}_I \mathcal{T}' \iff$ for all $X_0 a_1 X_1 \ldots X_n \in \mathcal{T}$ there is some $Y_0 a_1 Y_1 \ldots Y_n \in \mathcal{T}'$ with $X_i \supseteq Y_i$, for all $i \in 0..n$.

- $\mathcal{T} \leq_I^{lf} \mathcal{T}' \iff$ for all $X_0a_1X_1 \dots X_n \in \mathcal{T}$ there is some $Y_0a_1Y_1 \dots Y_n \in \mathcal{T}'$ with $X_n = Y_n$.
- $\mathcal{T} \leq_I^{lf\supseteq} \mathcal{T}' \iff$ for all $X_0a_1X_1 \dots X_n \in \mathcal{T}$ there is some $Y_0a_1Y_1 \dots Y_n \in \mathcal{T}'$ with $X_n \supseteq Y_n$.

Then, for each $\delta \in \{\supseteq, f, f\supseteq\}$ we write $p \leq_I^{l\delta} q$ if $LGO_I(p) \leq_I^{l\delta} LGO_I(q)$.

Since the definition of $\leq_I^{lf}$ ignores all the intermediate ready sets X_i with $i < n$ and requires the final ready sets to coincide, it is obvious that it defines the readiness preorder. Let us now prove that the two semantics based on failures are also characterized by our preorders $\leq_I^{lf\supseteq}$ and $\leq_I^{l\supseteq}$.

Proposition 4.18. *The preorder $\leq_I^{lf\supseteq}$ generates the failures preorder and $\leq_I^{l\supseteq}$ generates the failure trace preorder.*

Proof. The proof is based on the definition of initial failures of a process: we say that p rejects X if and only if $X \cap I(p) = \emptyset$. Then, $\langle \alpha, X\rangle$ is a failure of p if and only if $p \stackrel{\alpha}{\Longrightarrow} p'$ and p' rejects X. Using lgo's, for $\alpha = a_1 \dots a_n$, $\langle \alpha, X\rangle$ is a failure of p iff there exists $X_0a_1 \dots X_n \in \mathcal{T}$ such that $X_n \cap X = \emptyset$. Thus:

$$\begin{aligned} p \sqsubseteq_F p' &\iff \mathit{Failures}(p) \subseteq \mathit{Failures}(p') \\ &\iff \langle \alpha, X\rangle \in \mathit{Failures}(p') \text{ for all } \langle \alpha, X\rangle \in \mathit{Failures}(p) \\ &\iff X_0a_1 \dots X_n \in LGO_I(p) \text{ with } X_n \cap X = \emptyset \text{ implies that there exists} \\ &\qquad Y_0a_1 \dots Y_n \in LGO_I(p') \text{ with } Y_n \cap X = \emptyset\,, \end{aligned}$$

and then $p \leq_I^{lf\supseteq} p'$ implies $p \sqsubseteq_F p'$.

Conversely, assume that $p \sqsubseteq_F p'$ and recall that $p \leq_I^{lf\supseteq} p'$ iff for all $t = X_0a_1 \dots X_n \in LGO_I(p)$ there exists $Y_0a_1 \dots Y_n \in LGO_I(p')$ such that $X_n \supseteq Y_n$. For each set X, let us denote by X^c its complement. If $t \in LGO_I(p)$, we have $\langle \alpha, X_n^c\rangle \in \mathit{Failures}(p)$ and therefore $\langle \alpha, X_n^c\rangle \in \mathit{Failures}(p')$, which implies that there exists $p' \stackrel{\alpha}{\Longrightarrow} p''$ such that $I(p'') \cap X_n^c = \emptyset$. This means that there is some $t' = Y_0a_1 \dots a_nI(p'') \in LGO_I(p')$ with $I(p'') \subseteq X_n$, and therefore we can conclude that $p \leq_I^{lf\supseteq} p'$.

The proof for failure trace is very similar and we omit it. □

As a matter of fact, the characterization of failures by means of the reverse inclusion of offerings is not a great discovery at all: for instance, the same idea can be found in the definition of acceptance trees [29]. However, our sets of linear observations produce quite a nice characterization and allow us to forget about the notion of failures and consider instead reverse inclusion of offerings. But the most important property of our characterizations in terms of different orders on the set LGO_I is that they can be generalized to other local observation functions.

Definition 4.19. For $\mathcal{T}, \mathcal{T}' \subseteq LGO_N$ we define the orders $\leq_N^{l\supseteq}$, $\leq_N^{lf}$, and $\leq_N^{lf\supseteq}$ by:

- $\mathcal{T} \leq_N^{l\supseteq} \mathcal{T}' \iff$ for all $X_0a_1X_1 \dots X_n \in \mathcal{T}$ there is some $Y_0a_1Y_1 \dots Y_n \in \mathcal{T}'$ with $X_i \supseteq Y_i$ for all $i \in 0..n$.
- $\mathcal{T} \leq_N^{lf} \mathcal{T}' \iff$ for all $X_0a_1X_1 \dots X_n \in \mathcal{T}$ there is some $Y_0a_1Y_1 \dots Y_n \in \mathcal{T}'$ with $X_n = Y_n$.
- $\mathcal{T} \leq_N^{lf\supseteq} \mathcal{T}' \iff$ for all $X_0a_1X_1 \dots X_n \in \mathcal{T}$ there is some $Y_0a_1Y_1 \dots Y_n \in \mathcal{T}'$ with $X_n \supseteq Y_n$.

Then, for each $\delta \in \{\supseteq, f, f\supseteq\}$ we write $p \leq_N^{l\delta} q$ if $LGO_N(p) \leq_N^{l\delta} LGO_N(q)$.

By abuse of notation, we have used the superset inclusion symbol $\supseteq$ in the definitions above for any N. That is indeed the right interpretation for the cases $N = I, T$; however, for $N = U, C$ the superset inclusions degenerate to equalities while for $N = S$ it should be interpreted as $[\![p]\!]_S \geq_S [\![q]\!]_S$. Then, with the right notation we could have used such an inequality $[\![p]\!]_N \geq_N [\![q]\!]_N$ in all the cases.

When defining an observational semantics one expects the order between processes to be plain set inclusion as is the case, for instance, for the classic definition of failures semantics. Fortunately, it is easy to obtain such a characterization for the three semantics considered above by means of some suitable closure operators.

Definition 4.20. For $\mathcal{T} \subseteq LGO_N$, the following three closures are defined:

- $\overline{\mathcal{T}}^{\supseteq} = \{X_0 a_1 X_1 \dots a_n X_n \mid \text{there is some } Y_0 a_1 Y_1 \dots a_n Y_n \in \mathcal{T} \text{ with } X_i \supseteq Y_i \text{ for } i \in 0..n\}$.
- $\overline{\mathcal{T}}^{f} = \{X_0 a_1 X_1 \dots a_n X_n \mid \text{there is some } Y_0 a_1 Y_1 \dots a_n X_n \in \mathcal{T}\}$.
- $\overline{\mathcal{T}}^{f\supseteq} = \{X_0 a_1 X_1 \dots a_n X_n \mid \text{there is some } Y_0 a_1 Y_1 \dots a_n Y_n \in \mathcal{T} \text{ with } X_n \supseteq Y_n\}$.

Proposition 4.21. *All the operators in Definition 4.20 are indeed closures: if $\delta \in \{\supseteq, f, f\supseteq\}$ and $\mathcal{T}, \mathcal{T}' \subseteq LGO_N$, then $\mathcal{T} \subseteq \overline{\mathcal{T}}^{\delta}$ and $\overline{\overline{\mathcal{T}}^{\delta}}^{\delta} = \overline{\mathcal{T}}^{\delta}$; also, if $\mathcal{T} \subseteq \mathcal{T}'$ then $\overline{\mathcal{T}}^{\delta} \subseteq \overline{\mathcal{T}'}^{\delta}$.*

Proof. The first and third conditions are immediate from the definitions. As for the second, let $X_0 a_1 X_1 \dots a_n X_n \in \overline{\overline{\mathcal{T}}^{f}}^{f}$. Then, there exists $Y_0 a_1 Y_1 \dots a_n X_n \in \overline{\mathcal{T}}^{f}$ and thus there exists $Z_0 a_1 Z_1 \dots a_n X_n \in \mathcal{T}$, which implies $X_0 a_1 X_1 \dots a_n X_n \in \overline{\mathcal{T}}^{f}$; the inclusion in the other direction follows from monotonicity. Analogously for the other two operators. □

Proposition 4.22. *For all $\delta \in \{\supseteq, f, f\supseteq\}$, $\mathcal{T} \leq_N^{l\delta} \mathcal{T}'$ iff $\overline{\mathcal{T}}^{\delta} \subseteq \overline{\mathcal{T}'}^{\delta}$.*

Proof. It is easy but tedious, so only the case $\delta = f\supseteq$ is presented in detail. Assume $\mathcal{T} \leq_N^{lf\supseteq} \mathcal{T}'$: for all $t = X_0 a_1 X_1 \dots a_n X_n \in \mathcal{T}$ there exists $Y_0 a_1 Y_1 \dots a_n Y_n \in \mathcal{T}'$ with $X_n \supseteq Y_n$ and hence $t \in \overline{\mathcal{T}'}^{f\supseteq}$ and $\mathcal{T} \subseteq \overline{\mathcal{T}'}^{f\supseteq}$; $\overline{\mathcal{T}}^{f\supseteq} \subseteq \overline{\mathcal{T}'}^{f\supseteq}$ follows because of the properties of closures.

Conversely, from $\overline{\mathcal{T}}^{f\supseteq} \subseteq \overline{\mathcal{T}'}^{f\supseteq}$ it follows that $\mathcal{T} \subseteq \overline{\mathcal{T}'}^{f\supseteq}$ and thus for all $X_0 a_1 X_1 \dots a_n X_n \in \mathcal{T}$ there exists $Y_0 a_1 Y_1 \dots a_n Y_n \in \mathcal{T}'$ with $X_n \supseteq Y_n$: therefore $\mathcal{T} \leq_N^{lf\supseteq} \mathcal{T}'$. □

Definition 4.23. For each $\delta \in \{\supseteq, f, f\supseteq\}$, $p \in BCCSP$, and N a constraint, we define

$$LGO_N^{\delta}(p) = \overline{LGO_N(p)}^{\delta}.$$

Let us see which of the semantics in the spectrum are characterized by the orders $\leq_N^{l\delta}$ defined above.

Proposition 4.24. *For $N = U$ we have $\leq_U^{l} = \leq_U^{l\supseteq} = \leq_U^{lf} = \leq_U^{lf\supseteq} = \sqsubseteq_T$. As a consequence, the only semantics coarser than plain simulation that can be characterized by means of linear observations using L_U is the trace semantics.*

Proof. The first three equalities are obvious since U provides useless (empty) local information ($L_U = \{\cdot\}$). The last equality was proved in Proposition 4.15(1). □

Proposition 4.25. *For $N = C$ we have $\leq_C^l = \leq_C^{l\supseteq} = \leq_C^{lf} = \leq_C^{lf\supseteq} = \sqsubseteq_{CT}$. As a consequence, the only semantics coarser than complete simulation that can be characterized by means of linear observations using L_C is the completed trace semantics.*

Proof. Note that the local information at the intermediate steps of traces in LGO_C has to be *false*, since it corresponds to non-terminated states; thus, only the final states provide real information. Since in this case $\supseteq$ corresponds to Boolean equality, the first three equalities follow; the fourth was proved in Proposition 4.15(3). □

Proposition 4.26. *For $N = I$, $\leq_I^{lf\supseteq}$ characterizes the failures semantics, $\leq_I^{lf}$ the readiness semantics, $\leq_I^{l\supseteq}$ the failure trace semantics, and $\leq_I^{l}$ the ready trace semantics. Therefore, the possible worlds semantics is the only semantics in the ltbt spectrum coarser than ready simulation that cannot be characterized using lgo_I's.*

Proof. We have already proved (Propositions 4.15 and 4.18) the four characterizations, while $\sqsubseteq_{PW}$ cannot be characterized using lgo_I's because all the information available in our lgo_I's was needed to capture the ready trace semantic and it is well-known that the possible worlds semantics is strictly finer. □

As we will see in Section 4.3, the possible worlds semantics is the only deterministic branching semantics in the spectrum and will require the use of the deterministic branching observations introduced there to be characterized in an observational way. This is not the case, however, for the possible futures semantics (already discussed in [58]), and the impossible futures semantics [59].

Definition 4.27.

(1) The impossible futures semantics is defined as: $p \sqsubseteq_{IF} q$ if for all $S \subseteq \mathcal{P}(Act^*)$, if $p \stackrel{\alpha}{\Longrightarrow} p'$ with $T(p') \cap S = \emptyset$ then there exists $q \stackrel{\alpha}{\Longrightarrow} q'$ with $T(q') \cap S = \emptyset$.
(2) The possible futures semantics is defined as: $p \sqsubseteq_{PF} q$ if $p \stackrel{\alpha}{\Longrightarrow} p'$ then there exists $q \stackrel{\alpha}{\Longrightarrow} q'$ with $T(q') = T(p')$.

Proposition 4.28.

(1) *$\leq_T^{lf}$ is the possible futures preorder.*
(2) *$\leq_T^{lf\supseteq}$ is the impossible futures preorder.*

Proof.

(1) Obvious.
(2) Assume that $p \leq_T^{lf\supseteq} q$. Then $p \stackrel{\alpha}{\Longrightarrow} p'$, with $\alpha = a_1 \dots a_n$, implies $q \stackrel{\alpha}{\Longrightarrow} q'$ with $T(q') \subseteq T(p')$. Therefore, if $p \stackrel{\alpha}{\Longrightarrow} p'$ with $T(p') \cap X = \emptyset$ then $q \stackrel{\alpha}{\Longrightarrow} q'$ with $T(q') \cap X = \emptyset$ which implies $p \sqsubseteq_{IF} p'$.

Conversely, if $p \sqsubseteq_{IF} q$, $t = X_0 a_0 X_1 \dots X_n \in LGO_T(p)$ and $p \stackrel{\alpha}{\Longrightarrow} p'$ with $\alpha = a_1 \dots a_n$, obviously we have $T(p') \cap T(p')^c = \emptyset$, where $T(p')^c$ just represent the complement of the set $T(p')$. Now applying the definition of $\sqsubseteq_{IF}$, we have some $q \stackrel{\alpha}{\Longrightarrow} q'$ with $T(q') \cap T(p')^c = \emptyset$. Hence, there exists $t' = X'_0 a_0 X'_1 \dots X'_n \in LGO_T(q)$ with $T(q') \subseteq T(p')$, which implies $p \leq_T^{lf\supseteq} q$. □

As a matter of fact, the possible futures semantics is just below the 2-nested simulation semantics in the spectrum only because the trace simulation semantics is missing there.

At this point we are ready to present our first two "missing links", which arise through the remaining two orders: $\leq_T^l$ and $\leq_T^{l\supseteq}$.

Definition 4.29. The *possible futures trace semantics* is defined by lgo_T's related by $\leq_T^l$ and the *impossible futures trace semantics* is defined by $\leq_T^{l\supseteq}$.

Let us complete this part of the new extended spectrum by introducing the diamond generated by lgo_S's. This produces four new semantics coarser than 2-nested semantics. For instance, for the case of failures we obtain the following definition.

Definition 4.30. The *extended simulation failures* of a process p are defined as

$$\mathit{ExtSimFailures}(p) = \{\langle \alpha, p'' \rangle \mid \alpha \in A^*, p \stackrel{\alpha}{\Longrightarrow} p', p' \sqsubseteq_S p''\}.$$

The *simulation failures* of a process p are defined as $\mathit{SimFailures}(p) = \{\langle \alpha, B\rangle \mid p \stackrel{\alpha}{\Longrightarrow} p', B \cap BGO_U(p') = \emptyset\}$. We write $p \sqsubseteq_{SF} q$ iff $\mathit{SimFailures}(p) \subseteq \mathit{SimFailures}(q)$.

It can be proved that the inclusion $\mathit{SimFailures}(p) \subseteq \mathit{SimFailures}(q)$ holds if and only if $\mathit{ExtSimFailures}(p) \subseteq \mathit{ExtSimFailures}(q)$. Thus, simulation failures are essentially defined by translating the characterization of ordinary failures with the closure of readiness.

Proposition 4.31. $\leq_S^{lf\supseteq} = \sqsubseteq_{SF}$.

Proof. Analogous to the characterization of $\sqsubseteq_F$ in terms of $\leq_I^{lf\supseteq}$. □

4.3. **Deterministic branching observations.**

Definition 4.32.

(1) We say that a bgo is *deterministic* if the set of children $\{(a_i, bgo_i)\}$ of every node satisfies $a_i \neq a_j$ whenever $i \neq j$. We denote with $dBGO_N$ the set of deterministic observations in BGO_N.
(2) The set of deterministic branching observations (dbgo for short) of a process p is $dBGO_N(p) = BGO_N(p) \cap dBGO_N$.
(3) We write $p \leq_N^{db} q$ if $dBGO_N(p) \subseteq dBGO_N(q)$.

Like the linear observations, the set $dBGO_N(p)$ can be defined recursively and the corresponding semantics, compositionally.

Example 4.33. For the two processes $p = a(bc + bd)$ and $q = abc + abd$ we have that both deterministic observations in Figure 9 belong to $dBGO_I(p)$ and $dBGO_I(q)$. Indeed, that must be the case since it is easy to check that $dBGO_I(p) = dBGO_I(q)$.

In order to prove that dbgo's for the constraint I characterize the possible worlds semantics we first recall the definition of that semantics in [58].

Definition 4.34. A deterministic process p is a possible world of a process q if $p \sqsubseteq_{RS} q$. The set of possible worlds of p is denoted by $PW(p)$. We define the order $p \sqsubseteq_{PW} q$ iff $PW(p) \subseteq PW(q)$.

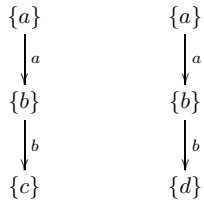


Figure 9: Deterministic branching observations.

When defining the possible worlds of a process we have to solve all the non-deterministic choices in it, each choice leading to one of its possible worlds. The same idea supports the selection of dbgo's to characterize this semantics: the non-deterministic branching observations in $BGO_N(p)$ are not present in $dBGO_N(p)$, where we have instead all the possible deterministic subtrees of every branching observation.

In our proof below we will relate the dbgo's in $dBGO_I(p)$ and the possible worlds in $PW(p)$. When necessary, we will consider observations in $dBGO_I(p)$ as processes in BCCSP by removing the information from their nodes; by abuse of notation we will also denote with *dbgo* the process obtained after such a removal. Also, we call *complete* those observations that, for every node labeled by an offering A, have a branch labeled by each of the actions in A.

Definition 4.35. The set of *complete deterministic* branching observations for the local observation function L_I is the set $cdBGO_I \subseteq dBGO_I$ recursively defined as:

- $\langle \emptyset, \emptyset \rangle \in cdBGO_I$.
- $\langle A, \{(a, cdbgo_a) \mid a \in A\} \rangle \in cdBGO_I$ for every $a \in A$ and $cdbgo_a \in cdBGO_I$.

For each $p \in$ BCCSP we define its set of complete deterministic branching observations $cdBGO_I(p) = dBGO_I(p) \cap cdBGO_I$.

We also associate to a deterministic process q its universal (complete deterministic) branching observation.

Definition 4.36. For a deterministic process p, its *universal deterministic* branching observation $cdbgo(p)$ is:

- $cdbgo(\mathbf{0}) = \langle \emptyset, \emptyset \rangle$.
- $cdbgo(\sum_{a \in A} ap_a) = \langle A, \{(a, cdbgo(p_a)) \mid a \in A\} \rangle$.

The following result is now immediate.

Proposition 4.37. *For every $p \in BCCSP$, $cdbgo(p) \in cdBGO_I(p)$.*

Lemma 4.38. *For every $q \in PW(p)$, $cdbgo(q) \in cdBGO_I(p)$.*

Proof. By structural induction on q:

- If q is $\mathbf{0}$, then $p \equiv \mathbf{0}$ and $\langle \emptyset, \emptyset \rangle \in cdBGO_I(\mathbf{0})$.
- If q is $\sum aq_a$, since $q \in PW(p)$ we have $q \sqsubseteq_{RS} p$. This implies $I(q) = I(p)$ and that, for all $a \in A$, there exists $p \xrightarrow{a} p_a$, $q_a \sqsubseteq_{RS} p_a$, so that $q_a \in PW(p_a)$. By induction hypothesis, $cdbgo(q_a) \in cdBGO_I(p)$. Now, by definition, $cdbgo(q) = \langle A, \{(a, cdbgo(q_a)) \mid a \in A)\} \rangle$

and, from $p \xrightarrow{a} p_a$ and $I(p) = I(q)$, we conclude $cdbgo(q) \in dBGO_I(p)$ and therefore $cdbgo(q) \in cdBGO_I(p)$. □

Lemma 4.39. *For every process q such that $cdbgo(q) \in cdBGO_I(p)$ we have $q \sqsubseteq_{RS} p$ and therefore $q \in PW(p)$.*

Proof. We will prove that the set $S = \{(q,p) \mid cdbgo(q) \in cdBGO_I(p)\}$ is a ready simulation. Obviously, for $(q,p) \in S$ it is $I(q) = I(p)$ and, if $q \xrightarrow{a} q_a$, there exists $p \xrightarrow{a} p_a$ with $cdbgo(q_a) \in cdBGO_I(p_a)$, which shows that $(q_a, p_a) \in S$ and that S is a ready simulation. □

Theorem 4.40. *For all processes $p_1, p_2 \in BCCSP$, $p_1 \sqsubseteq_{PW} p_2$ iff $p_1 \leq_I^{db} p_2$.*

Proof. ($\Leftarrow$) For $q \in PW(p_1)$, by Lemma 4.38 we have $cdbgo(q) \in cdBGO_I(p_1)$ and therefore $cdbgo(q) \in cdBGO_I(p_2)$. Now, by Lemma 4.39, $q \sqsubseteq_{RS} p_2$ and thus $q \in PW(p_2)$.

($\Rightarrow$) Let $dbgo \in dBGO_I(p_1)$: by definition of $dBGO_I(p_1)$ it is clear that we can extend $dbgo$ into some $dbgo' \in cdBGO_I(p_1)$. Now, by Lemma 4.39, $dbgo' \sqsubseteq_{RS} p_1$ (taking $dbgo'$ as a deterministic process). Therefore, $dbgo' \in PW(p_1)$ and thus $dbgo' \in PW(p_2)$ and, by Lemma 4.38, $cdbgo(dbgo') = dbgo' \in cdBGO_I(p_2)$: hence $dbgo \in dBGO_I(p_2)$ as required. □

Remark 4.41. If we consider infinite processes, then our characterization of $\sqsubseteq_{PW}$ by means of $\leq_I^{db}$ only works if we restrict ourselves to image-finite processes. We will continue the discussion on this part when studying the logical characterization of this semantics at Section 6.

Let us briefly consider the remaining new semantics definable by means of deterministic branching observations. It is clear that in all cases the corresponding orders verify $\leq_N^b \subseteq \leq_N^{db} \subseteq \leq_N^l$, so that the associated semantics will be situated between the corresponding semantics defined by branching observations in BGO_N and linear observations in LGO_N, as is the case for the possible worlds semantics, located between the ready simulation semantics and the ready trace semantics.

Admittedly, most of these semantics are rather strange and this is probably the reason why, as far as we know, they have not been previously considered. However, the simplest of them all, that corresponding to $N = U$, has properties similar to the possible worlds semantics and, in fact, can be defined by simply removing from its definition the "R" in the condition $q \sqsubseteq_{RS} p$. Hence, we can regard as possible worlds those deterministic implementations where we offer just a part of the action offered by the given process.

Definition 4.42. The partial possible worlds of a process p are those deterministic processes that verify $q \sqsubseteq_S p$. We denote with $PW_U(p)$ the set of partial possible worlds of a process p and define $p \sqsubseteq_{UPW} q$ if $PW_U(p) \subseteq PW_U(q)$.

Proposition 4.43. *For all processes $p_1, p_2 \in BCCSP$, $p_1 \sqsubseteq_{UPW} p_2$ iff $p_1 \leq_U^{db} p_2$.*

Proof. Similar to Theorem 4.40, simplified by the fact that all $dbgo$ in $PW_U(p)$ satisfy $dbgo \sqsubseteq_S p$. □

Example 4.44. We have $a \sqsubseteq_{UPW} a + b$ since $\langle \cdot, \{(a, \emptyset)\}\rangle \in dBGO_U(a+b)$. By contrast, for $p = ab + ac$ and $q = a(b+c)$ we have $p \sqsubseteq_{UPW} q$ but $q \not\sqsubseteq_{UPW} p$ because $\langle \cdot, \{(a, \langle \cdot, \{(b, \langle \cdot, \emptyset\rangle), (c, \langle \cdot, \emptyset\rangle)\}\rangle)\}\rangle \in dBGO_U(q) - dBGO_U(p)$.

Analogously, for any other constraint N we could define the N-possible worlds order $\sqsubseteq_{NPW}$ using $\sqsubseteq_{NS}$ instead of $\sqsubseteq_S$ at Definition 4.42. However, it is easy to see that when N is fine enough, e.g. $N = T$, this order would become totally wrong. Instead, we can

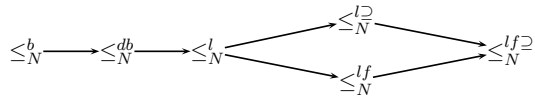


Figure 10: Basic layer in the linear time-branching time spectrum.

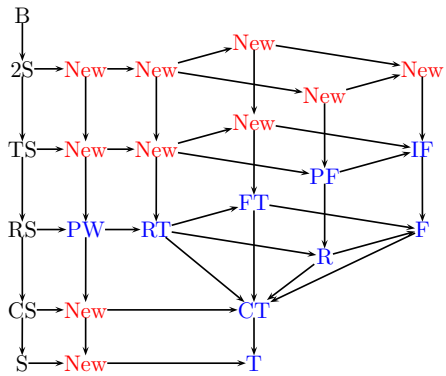


Figure 11: Semantics in the new linear time-branching time spectrum.

still consider the observations in $dBGO_N$ and by means of them we define the "reasonable" deterministic branching semantics, for any layer in the spectrum.

The extended spectrum can now be depicted as in Figures 11 and 10.

4.4. **Back to branching observations.** The orders $\leq_N^{l\delta}$ with $\delta \in \{\supseteq, f, f\supseteq\}$ that characterize some of the linear semantics studied in Section 4.2, restricted in several ways the use of the local information, when characterizing those semantics. The same scheme can be generalized to the branching observations. This way, for each constraint N we would obtain three new branching semantics based on bgo's in BGO_N which, together with the original N-simulation semantics, would constitute a diamond of branching semantics at a higher layer in our extended ltbt spectrum. The introduction of these new semantics also offers a clearer view of the spectrum, with two main levels of branching and linear semantics and an intermediate one of deterministic branching semantics. Although this provides the means for obtaining a host of new semantics, it is also true that most of them are bizarre, in sharp contrast with the fact that the corresponding orders gave rise to interesting semantics when applied to linear observations.

To illustrate the comments above, next we consider in some detail the case $N = I$, which corresponds to the most interesting semantics.

Definition 4.45. For $bgo, bgo' \in BGO_I$ we define:

- $bgo \leq_I^{\supseteq} bgo' \iff$ $\big(bgo = \langle A_1, S_1\rangle$ and $bgo' = \langle A_2, S_2\rangle$ and $A_1 \supseteq A_2$ and $S_1 = \{(a_i, bgo_i) \mid i \in I\}$ and $S_2 = \{(a_i, bgo'_i) \mid i \in I\}$ and for all $i \in I$ $(bgo_i \leq_I^{\supseteq} bgo'_i)\big)$.

- $bgo \leq_I^f bgo' \iff \big(bgo = \langle A_1, \emptyset\rangle$ and $bgo' = \langle A_1, \emptyset\rangle\big)$ or $\big(bgo = \langle A_1, S_1\rangle$ and $bgo' = \langle A_2, S_2\rangle$ and $S_1 = \{(a_i, bgo_i) \mid i \in I\}$ and $S_2 = \{(a_i, bgo_i') \mid i \in I\}$ and for all $i \in I$ $(bgo_i \leq_I^f bgo_i')\big)$.
- $bgo \leq_I^{f\supseteq} bgo' \iff \big(bgo = \langle A_1, \emptyset\rangle$ and $bgo' = \langle A_2, \emptyset\rangle$ and $A_1 \supseteq A_2\big)$ or $\big(bgo = \langle A_1, S_1\rangle$ and $bgo' = \langle A_2, S_2\rangle$ and $S_1 = \{(a_i, bgo_i) \mid i \in I\}$ and $S_2 = \{(a_i, bgo_i') \mid i \in I\}$ and for all $i \in I$ $(bgo_i \leq_I^{f\supseteq} bgo_i')\big)$.

Definition 4.46. For $\mathcal{B}, \mathcal{B}' \subseteq BGO_I$ and $\delta \in \{\supseteq, f, f\supseteq\}$, we define the orders $\leq_I^{b\delta}$ by:
- $\mathcal{B} \leq_I^{b\delta} \mathcal{B}' \iff$ for all $bgo \in \mathcal{B}$ there exists $bgo' \in \mathcal{B}'$ with $bgo \leq_I^{\delta} bgo'$.

Then, we write $p \leq_I^{b\delta} q$ if $BGO_I(p) \leq_I^{b\delta} BGO_I(q)$.

It is somewhat surprising to discover that $\leq_I^{b\supseteq} = \leq_I^b$, since this was not the case for their linear "projections" $\leq_I^{l\supseteq}$ and $\leq_I^l$.

Proposition 4.47. *For all processes $p_1, p_2 \in BCCSP$, $p_1 \leq_I^{b\supseteq} p_2$ iff $p_1 \leq_I^b p_2$.*

Proof. Assume that $p_1 \leq_I^{b\supseteq} p_2$ and let $bgo \in BGO_I(p_1)$: it is clear that it can be extended into a complete $cbgo \in BGO_I(p_1)$. Then, there exists some $cbgo' \in BGO_I(p_2)$ with $cbgo \leq_I^{\supseteq} cbgo'$ and, since $cbgo$ is complete, $cbgo = cbgo'$ and hence $bgo \in BGO_I(p_2)$. The other implication is trivial. □

Example 4.48. For $p_1 = a(b + c)$ and $p_2 = ab + ac$, $p_1 \leq_I^{l\supseteq} p_2$ but $p_1 \not\leq_I^l p_2$. However, $p_1 \not\leq_I^{b\supseteq} p_2$ since for $bgo = \langle\{a\}, (a, \langle\{b, c\}, \{(b, \emptyset), (c, \emptyset)\}\rangle)\rangle \in BGO_I(p_1)$ there is no $bgo' \in BGO_I(p_2)$ with $bgo \leq_I^{l\supseteq} bgo'$.

By contrast, the branching semantics defined by $\leq_I^{bf}$ and $\leq_I^{bf\supseteq}$ are indeed new.

Example 4.49. For the processes p and q in Figure 12, $p \leq_I^{bf} q$ but $p \not\leq_I^b q$.

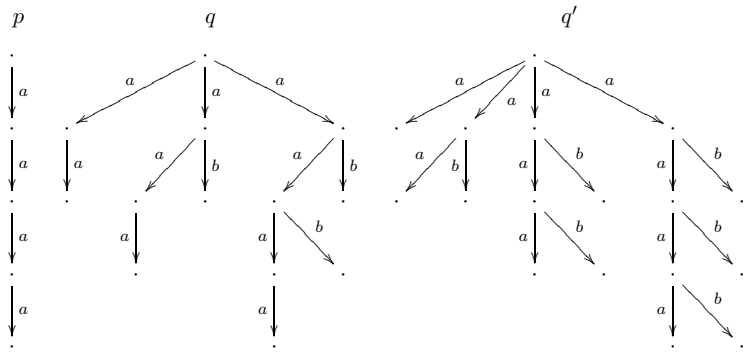


Figure 12: Three processes.

This example shows that it is quite difficult to characterize this semantics as a simulation one. Furthermore, we conjecture that it is not finitely axiomatizable in the classic way (that means using only unconditional axioms). As a matter of fact, we were also unable to find any conditional axiomatization, what we interpret as a "proof" of the fact that these new branching semantics are quite strange.

Definition 4.50. We say that $R \subseteq BGO_I \times \mathrm{BCCSP}$ is a *final-ready simulation* when:

- $(\langle A, \emptyset\rangle, q) \in R$ implies $I(q) = A$.
- $(\langle A, \{(a_i, bgo_i)\}\rangle, q) \in R$ implies that for all $i \in I$ there exists $q \xrightarrow{a_i} q_i$ such that $(bgo_i, q_i) \in R$.

We say that p is final-ready simulated by q when for all $bgo \in BGO_I(p)$ there exists a final-ready simulation R with $(bgo, q) \in R$, and write $p \sqsubseteq_{fRS} q$.

Theorem 4.51. *For all $p, q \in BCCSP$, $p \sqsubseteq_{fRS} q$ iff $p \leq_I^{bf} q$.*

Example 4.52. It is easy to check that for p and q' as in Figure 12 we have $p \leq_I^{bf\supseteq} q'$ but $p \not\leq_I^{bf} q'$.

Final failure simulations are defined exactly like final-ready simulations but substituting $I(q) \subseteq Act$ for $I(q) = A$ in the first clause, giving rise to the order $\sqsubseteq_{fFS}$ between processes.

Theorem 4.53. *For all $p, q \in BCCSP$, $p \sqsubseteq_{fFS} q$ iff $p \leq_I^{bf\supseteq} q$.*

As previously noted, these are certainly bizarre semantics but we believe it is interesting to indicate their existence because, by analogy to the linear case, their definitions in terms of branching observations look quite natural. However, it also seems that when dealing with branching observations the introduction of any kind of asymmetry in the treatment of local observations produces quite involved semantics.

5. Relating the observational and equational frameworks

In this section we tie up all loose ends and show how our unification theory is fully self-contained. Namely, we prove the results on axiomatic characterizations in Section 3 from the observational semantics developed in Section 4: we show how the equations are deduced from the observations in a general way without resorting to the already existing axiomatizations.

One of the key points of this section is to illustrate how the particular proofs needed in Section 3 for every one of the semantics can be replaced by a generic proof that stands for a whole family of semantics. In fact, we will show in Section 8 that the same proof is still valid for the new semantics suggested in Roscoe's work.

5.1. Semantics coarser than ready simulation.

Let us now see how, from this uniform definition of the linear semantics, the proofs of the correctness and completeness of the corresponding axiomatizations can be derived in a uniform way avoiding the case analyses of Sections 3.1 and 3.2. Although this could be done generically, with $N \in \{U, C, I, T\}$, we prefer to start with the particular case $N = I$, which corresponds to the most popular semantics already studied in Section 3.1.

To start with, we show how the axiomatizations can be synthetized from the observational characterizations. Our general axiom (*ND*) for the reduction of non-determinism

specifies the hypothesis $M(x,y,w)$ under which the process $ax + a(y+w)$ can be (syntactically) expanded by adding a new summand $a(x+y)$ without changing its semantics. Then, let us compare the two sides of our general axiom. Since $I(ax+a(y+w)) = I(ax) = I(a(y+w)) = \{a\}$, we have

$$\begin{array}{lcl} LGO_I(ax+a(y+w)) & = & LGO_I(ax) \cup LGO_I(a(y+w)), \\ LGO_I(a(y+w)) & = & \{\langle\{a\}\rangle\} \cup \\ & & \{\langle\{a\},a,I(y+w)\rangle \circ S \mid \langle\{a\},a,I(y)\rangle \circ S \in LGO_I(ay) \vee \\ & & \qquad \langle\{a\},a,I(w)\rangle \circ S \in LGO_I(aw)\}. \end{array}$$

Notice then that the observations of $a(y+w)$ are exactly those of $ay+aw$ simply replacing $I(y)$ or $I(w)$, respectively, by $I(y+w) = I(y) \cup I(w)$. Analogously,

$$\begin{array}{lcl} LGO_I(a(x+y)) & = & \{\langle\{a\}\rangle\} \cup \\ & & \{\langle\{a\},a,I(x+y)\rangle \circ S \mid \langle\{a\},a,I(x)\rangle \circ S \in LGO_I(ax) \vee \\ & & \qquad \langle\{a\},a,I(y)\rangle \circ S \in LGO_I(ay)\}. \end{array}$$

Now, in order to get the adequate condition $M_Z(x,y,w)$ for each of the semantics, let us examine the formulas that define the preorders $\preceq_I^{lY}$:

- $\preceq_I^{l}$. To have $LGO_I(a(x+y)) \subseteq LGO_I(ax) \cup LGO_I(a(y+w))$ it is enough to require $\{\langle\{a\},a,I(x)\cup I(y)\rangle \circ S \mid \langle I(x)\rangle \circ S \in LGO_I(x)\} \subseteq \{\langle\{a\},a,I(x)\rangle \circ S \mid \langle I(x)\rangle \circ S \in LGO_I(x)\}$ and $\{\langle\{a\},a,I(x)\cup I(y)\rangle \circ S \mid \langle I(y)\rangle \circ S \in LGO_I(y)\} \subseteq \{\langle\{a\},a,I(y)\cup I(w)\rangle \circ S \mid \langle I(y)\rangle \circ S \in LGO_I(y)\}$. Thus, a first proposal for M_{RT} would be
$$I(y) \subseteq I(x) \wedge I(x) = I(y) \cup I(w).$$
However, due to the fact that this axiom will be used in combination with (RS), the following, more restrictive but simpler form, can be used instead:
$$M_{RT}(x,y,w) \iff I(x) = I(y) \wedge I(w) \subseteq I(y).$$
Clearly, this form is stronger than the condition synthetized above. Reciprocally, $a(x+y) \preceq ax+a(y+w)$ can be proved from the assumptions $I(y) \subseteq I(x)$ and $I(x) = I(y) \cup I(w)$ using (RS) first to get $a(x+y) \preceq a(x+y+w)$, and then (ND) instantiated with M_{RT} to obtain $a(x+y+w) \preceq ax + a(y+w)$.
- $\preceq_I^{l\supseteq}$. We need the inclusion $\overline{LGO_I(a(x+y))}^{\supseteq} \subseteq \overline{LGO_I(ax+a(y+w))}^{\supseteq}$ to hold. Since $I(x) \cup I(y) \supseteq I(x)$, the general observations in $a(x+y)$ that arise from x will be also in $\overline{LGO_I(ax)}^{l\supseteq}$. For those that arise from y, it is required that $I(x) \cup I(y) \supseteq I(y) \cup I(w)$. Once again, (RS) can be used to simplify this condition into the simpler
$$M_{FT}(x,y,w) \iff I(w) \subseteq I(y).$$
The less restrictive variant of the axiom can be derived from the stronger one and (RS) as follows. Taking $w = \mathbf{0}$, since $I(\mathbf{0}) \subseteq I(y)$ we obtain $a(x+y) \preceq ax + ay$ from (ND^{FT}); in particular, $a(x+y+w) \preceq ax + a(y+w)$. Also, by (RS), $x + y \preceq (x+y) + (x+y+w)$, from where it follows $a(x+y) \preceq a(x+y+w)$.
- $\preceq_I^{lf}$. We consider the inclusion $\overline{LGO_I(a(x+y))}^{f} \subseteq \overline{LGO_I(ax+a(y+w))}^{f}$. We only have to consider the lgo $\langle\{a\},a,I(x)\cup I(y)\rangle$ in $\overline{LGO_I(a(x+y))}^{f}$ and show that it also belongs to $\overline{LGO_I(ax+a(y+w))}^{f}$, since all lgo's of length greater than 1 start with the prefix $\langle\{a\},a\rangle$. For that, either $I(x) \cup I(y) = I(x)$ or $I(x) \cup I(y) = I(y) \cup I(w)$, that is,

$I(y) \subseteq I(x)$ or $I(x) \cup I(y) = I(y) \cup I(w)$. Again, we can remove the second condition and define

$$M_R(x, y, w) \iff I(y) \subseteq I(x)$$

since, whenever $I(x) \cup I(y) = I(y) \cup I(w)$, $a(x + y + w) \preceq ax + a(y + w)$ can be obtained by taking $x = y + w$, $y = x$, and $w = \mathbf{0}$, and then by applying (RS) we conclude $a(x + y) \preceq ax + a(y + w)$.

- $\leq_I^{lf\supseteq}$. An argument analogous to the previous one leads us to check that $I(x) \cup I(y) \supseteq I(x)$ or $I(x) \cup I(y) \supseteq I(y) \cup I(w)$, and the first is certainly true.

In order to prove the completeness of our axiomatizations we introduce the following notions of head normal forms.

Definition 5.1. For $p = \sum_{a \in X_0} \sum_{i \in I_a} a p_a^i$ and $Z \in \{F, R, FT, RT\}$, its Z-head normal form $hnf^Z(p)$ is:

- For $a \in X_0$, $i \in I_a$, and $X_1 \subseteq \bigcup_{i \in I_a} I(p_a^i)$ such that $I(p_a^i) \subseteq X_1$, we define $hnf^Z(p, a, i, X_1) = a(p_a^i + \sum\{p_a^j|_{X_1} \mid j \neq i, M_Z(p_a^i, p_a^j|_{X_1}, p_a^j|_{\overline{X_1}})\})$.
- $hnf^Z(p) = p + \sum_{a \in X_0} \sum_{i \in I_a} \sum_{X_1 \subseteq \bigcup I(p_a^i)} hnf^Z(p, a, i, X_1)$.

It is clear that several redundancies arise in this definition: for example, if $Z = RT$ then $hnf^Z(p, a, i, X_1) = hnf^Z(p, a, i, I(p_a^i))$, so that the argument X_1 would not be needed in this case. We prefer to maintain the generic definition in order to allow a homogeneous treatment of all the semantics.

Proposition 5.2. *For $Z \in \{RT, FT, R, F\}$, $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash hnf^Z(p) \preceq p$.*

Proof. Let $p = \sum_{a \in X_0} \sum_{i \in I_a} a p_a^i$.

Considering the definition of $hnf^Z(p, a, i, X_1)$, let us consider an enumeration of the set of j's contributing to it: If $J_i = \{j \neq i \mid M_Z(p_a^i, p_a^{j_1}|_{X_1}, p_a^{j_1}|_{\overline{X_1}})\}$, we take $\{j_k \mid k = 1 \ldots |J_i|\} = J_i$.

Then we can prove by induction on l that for all $0 \leq l \leq |J_i|$ we have $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash a(p_a^i + \sum_{h=1}^{l} p_a^{j_h}|_{X_1}) \preceq a p_a^i + \sum_{h=1}^{l} a p_a^{j_h}$.

The case of $l = 0$ is trivial. Assuming the result for l, we prove the result for $l + 1$. From $M_Z(p_a^i, p_a^{j_{l+1}}|_{X_1}, p_a^{j_{l+1}}|_{\overline{X_1}})$ we can infer $M_Z(p_a^i + \sum_{h=1}^{l} p_a^{j_h}|_{X_1}, p_a^{j_{l+1}}|_{X_1}, p_a^{j_{l+1}}|_{\overline{X_1}})$ so that we can derive $\vdash a(p_a^i + \sum_{h=1}^{l+1} p_a^{j_h}|_{X_1}) \preceq a(p_a^i + \sum_{h=1}^{l} p_a^{j_h}) + a p_a^{j_{l+1}}$; and applying the i.h. we conclude $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash a(p_a^i + \sum_{h=1}^{l+1} p_a^{j_h}|_{X_1}|_{X_1}) \preceq a p_a^i + \sum_{h=1}^{l+1} a p_a^{j_h}$.

From this we immediately obtain $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash hnf^Z(p, a, i, X_1) \preceq p|_a$. Finally, adding all these inequalities we conclude $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash hnf^Z(p) \preceq p$. □

Let us define $l(F) = lf\supseteq$, $l(FT) = l\supseteq$, $l(R) = lf$, and $l(RT) = l$. In order to apply structural induction to prove the completeness of the axiomatizations we need to show that, whenever $p = \sum_{a \in X_0} \sum_{i \in I_a} a p_a^i$ and $p \leq_I^{l(Z)} q$, there is a summand $a h_a^k$ of $hnf^Z(q)$ such that $p_a^i \leq_I^{l(Z)} h_a^k$ for each $a \in Act$, $i \in I_a$.

Proposition 5.3. *Let $Z \in \{F, FT, R, RT\}$, and let $p = \sum_{a \in X_0} \sum_{i \in I_a} a p_a^i$, and $q = \sum_{a \in X_0} \sum_{j \in J_a} a q_a^j$. If $p \leq_I^{l(Z)} q$ then there exists a summand $a h_a^k$ of $hnf^Z(q)$ such that $p_a^i \leq_I^{l(Z)} h_a^k$.*

Proof. Using Definition 4.19 and Proposition 4.22, we need to show that there exists $\overline{LGO_I(p^i_a)}^{l(Z)} \subseteq \overline{LGO_I(h^k_a)}^{l(Z)}$ but, due to the fact that $\overline{(\_)}^{l(Z)}$ is a closure operator (Proposition 4.21), it is enough to prove that $LGO_I(p^i_a) \subseteq \overline{LGO_I(h^k_a)}^{l(Z)}$. For $\langle I(p^i_a)\rangle \in LGO_I(p^i_a)$, since $p \leq_I^{l(Z)} q$ there is some q^j_a such that $\langle I(p^i_a)\rangle \in LGO_I(q^j_a)^{l(Z)}$; we then consider $hnf^Z(q, a, j, I(p^i_a)) = ah^k_a$.

If $t \in LGO_I(p^i_a)$ then $\langle I(p), a\rangle \circ t \in LGO_I(p) \subseteq \overline{LGO_I(q)}^{l(Z)}$ and there exists j_t such that $t \in \overline{LGO_I(q^{j_t}_a)}^{l(Z)}$. In addition, $M_Z(q^j_a, q^{j_t}_a|_{I(p^i_a)}, q^{j_t}_a|_{\overline{I(p^i_a)}})$:

- If $Z = RT$, then $t \in LGO_I(q^{j_t}_a)$ and therefore $I(q^{j_t}_a) = I(p^i_a) = I(q^j_a)$. Hence, condition $M_{RT}(q^j_a, q^{j_t}_a|_{I(p^i_a)}, \mathbf{0})$ holds and therefore $M_{RT}(q^j_a, q^{j_t}_a|_{I(p^i_a)}, q^{j_t}_a|_{\overline{I(p^i_a)}})$.
- If $Z = FT$, from $t \in \overline{LGO_I(q^{j_t}_a)}^{\supseteq}$ it follows that $I(q^{j_t}_a) \subseteq I(p^{j_t}_a)$ and therefore $I(q^{j_t}_a|_{\overline{I(p^i_a)}}) = \emptyset \subseteq I(q^{j_t}_a|_{I(p^i_a)})$. Hence, $M_{FT}(q^j_a, q^{j_t}_a|_{I(p^i_a)}, q^{j_t}_a|_{\overline{I(p^i_a)}})$.
- If $Z = R$, from $\langle I(p^i_a)\rangle \in \overline{LGO_I(q^j_a)}^{f}$ we have that $I(p^i_a) = I(q^j_a)$ and thus $I(q^{j_t}_a|_{I(p^i_a)}) \subseteq I(q^j_a)$ and $M_R(q^j_a, q^{j_t}_a|_{I(p^i_a)}, q^{j_t}_a|_{\overline{I(p^i_a)}})$.
- For $Z = F$ it is trivial since $M_F(x, y, w)$ is always true.

Therefore $q^{j_t}_a$ is one of the summands of h^k_a and, since $t \in \overline{LGO_I(q^{j_t}_a)}^{l(Z)}$, we have $p^i_a \leq_I^{l(Z)} h^k_a$. □

Theorem 5.4 (Soundness and completeness)**.** *For $Z \in \{RT, FT, R, F\}$:*

$$p \leq_I^{l(Z)} q \text{ iff } \{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash p \preceq q.$$

Proof. (Soundness) The axiomatizations are sound because of the way they have been derived.

(Completeness) By structural induction on p.

- Let p be $\mathbf{0}$. As usual, we can consider terms up to bisimulation since B_1–B_4 are equations needed for all the semantics. If $p \leq_I^{l(Z)} q$, then q must be $\mathbf{0}$ (or bisimilar to $\mathbf{0}$) because the set of local observations of $\mathbf{0}$ is empty and cannot contain any observations (see Definition 4.19.)
- If $p = \sum_{a\in X_0} \sum_{i\in I_a} ap^i_a$ then, by Proposition 5.3, $p \leq_I^{l(Z)} q$ implies that there exists a summand ah^k_a of $hnf^Z(q)$ such that $p^i_a \leq_I^{l(Z)} h^k_a$. By induction hypothesis, $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash p^i_a \preceq h^k_a$ and therefore $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash ap^i_a \preceq ah^k_a$; adding all these inequalities and using (RS), which is allowed because $I(p) = I(q)$, it follows that $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash p \preceq hnf^Z(q)$ and, by Proposition 5.2, $\{B_1\text{–}B_4, (RS), (ND^Z)\} \vdash p \preceq q$. □

5.2. **The semantics that are not coarser than ready simulation.** Once we have a clear picture of the semantics that are coarser than ready simulation, it is time to consider the rest of the semantics in the spectrum. Let us start with the possible futures and the impossible futures semantics. Recall that we have shown that they can be described by LGO_T observations so that they are defined by $\leq_T^{lf\supseteq}$ and $\leq_T^{lf}$, respectively.

We introduce the T-versions of our (ND^Z) axioms: all of them are instances of our general axiom for the reduction of non-determinism and therefore are defined by the adequate

constraint $M_Z^T(x,y,w)$. As expected, they are obtained by substituting every occurrence of I in $M_Z(x,y,w)$ by the observer T defining the traces of processes.

Definition 5.5. The constraints M_Z^T that characterize the semantics coarser than T-simulation semantics are:

$$\begin{array}{ll}
(T\text{-}ND^F) & M_F^T(x,y,w) \iff \text{true} \\
(T\text{-}ND^R) & M_R^T(x,y,w) \iff T(x) \supseteq T(y) \\
(T\text{-}ND^{FT}) & M_{FT}^T(x,y,w) \iff T(w) \subseteq T(y) \\
(T\text{-}ND^{RT}) & M_{RT}^T(x,y,w) \iff T(x) = T(y) \text{ and } T(w) \subseteq T(y)
\end{array}$$

As indicated in Section 4.2 (Definition 4.29), the semantics associated to the last two conditions do not appear in the ltbt spectrum and, as far as we know, they have not been previously studied nor even defined.

Using the same arguments as in Section 5.1, we can prove that $\leq_T^{l(Z)}$ satisfies the axiom $(T\text{-}ND^Z)$ for $Z \in \{RT, FT\}$.

Proposition 5.6. *$M_Z^T(x,y,w)$ implies $T(a(x+y)) = T(ax+a(y+w))$ for $Z \in \{RT, FT\}$. However, this is not the case for $Z \in \{R, F\}$*

Proof. M_{RT}^T implies M_{FT}^T, and therefore $T(y+w) = T(y)$, which leads to $T(ax+a(y+w)) = T(a(x+y))$. Neither M_R^T nor M_F^T refer to w and therefore, in general, $T(ax+a(y+w)) \neq T(a(x+y))$ in those cases. □

Note that when proving the correctness of the corresponding axiom (ND^Z) for $\leq_I^{l(Z)}$ we had $I(a(x+y)) = \{a\} = I(ax+a(y+w))$ in all cases. Now, $T(a(x+y)) = T(ax+a(y+w))$ only under the constraints corresponding to the finer semantics $\sqsubseteq_{FT}$ and $\sqsubseteq_{RT}$. The properties of the prefixes appearing in all the terms in both sides of the axiom (ND) are not used anymore in the proofs in Section 5.1, so they can be transferred to the T-semantics, thus proving the correctness of $(T\text{-}ND^Z)$ for both $\leq_T^{l(RT)}$ and $\leq_T^{l(FT)}$.

The introduction of the equational version $(ND_{\equiv})$ of the axiom (ND) now becomes crucial in order to preserve the generality of our unifying work. We saw that under (RS) these two axioms were equivalent. However, when observing the set of traces $T(x)$ of any process, instead of just the initial offer $I(x)$ we need to consider T-simulations, that are constrained by the condition $T(x,y) \iff T(x) = T(y)$; under the corresponding axiom (TS), things turn out to be different.

Proposition 5.7. $T(a(x+y)+ax+a(y+w)) = T(ax) \cup T(ay) \cup T(aw) = T(ax+a(y+w))$.

As a consequence, for $(T\text{-}ND_+^Z)$ and $(T\text{-}ND_{\equiv})$ we can apply the same arguments used in Section 5.1 to show that (ND^Z) was satisfied by $\leq_I^{l(Z)}$.

Proposition 5.8. *For $Z \in \{RT, FT, R, F\}$, the preorder $\leq_T^{l(Z)}$ satisfies the axiom $(T\text{-}ND_+^Z)$ and also $(T\text{-}ND_{\equiv}^Z)$.*

Proof. To show that $\leq_T^{l(Z)}$ satisfies $(T\text{-}ND_+^Z)$ we just need to apply Proposition 5.7 and follow the line of thought in the second bullet on page 37, substituting the observer T for I. For the other axiom, from $T(a(x+y)) \subseteq T(ax+a(y+w))$ it follows that $\{(TS)\} \vdash ax+a(y+w) \preceq (ax+a(y+w))+a(x+y)$. □

Notice that for $Z \in \{RT, FT\}$ we can also obtain the correctness of $(T\text{-}ND^Z_{\equiv})$ from that of $(T\text{-}ND^Z)$ and vice versa, as a consequence of the following fact.

Proposition 5.9. *The axiomatization* $\{B_1\text{–}B_4, (TS), (T\text{-}ND^Z)\}$ *is equivalent to the axiomatization* $\{B_1\text{–}B_4, (TS), (T\text{-}ND^Z_{\equiv})\}$ *for* $Z \in \{RT, FT\}$.

Proof. Let us first show that $\{B_1\text{–}B_4, (TS), (T\text{-}ND^Z)\}$ is equivalent to $\{B_1\text{–}B_4, (TS), (T\text{-}ND^Z_{+})\}$. This holds because $(T\text{-}ND^Z)$ implies $(T\text{-}ND^Z_{+})$ and, since $T(w) \subseteq T(y)$ implies $T(a(x+y)) = T(ax + a(y+w))$ and then we have $\{B_1\text{–}B_4, (TS)\} \vdash a(x+y) \preceq a(x+y) + (ax + a(y+w))$.

To prove $\{B_1\text{–}B_4, (TS), (T\text{-}ND^Z_{+})\}$ equivalent to $\{B_1\text{–}B_4, (TS), (T\text{-}ND^Z_{\equiv})\}$ we only need to show that $\{B_1\text{–}B_4, (TS), (T\text{-}ND^Z_{+})\} \vdash (M^T_Z(x,y,w) \Rightarrow ax + a(y+w) \preceq ax + a(y+w) + a(x+y))$, but we have that for $Z \in \{RT, FT\}$, $M^T_Z(x,y,w)$ implies $T(w) \subseteq T(y)$, so that $T(a(x+y)) = T(ax + a(y+w))$ and therefore $\{(TS)\} \vdash ax + a(y+w) \preceq (ax + a(y+w)) + a(x+y)$. □

The important fact about the obtained sets of correct axioms for the semantics $\leq_T^{l(Z)}$ is that, although our proofs of completeness for the axiomatizations $\{B_1\text{–}B_4, (RS), (ND^Z)\}$ considered the inequational axioms (ND^Z), they are also valid for the axiomatizations $\{B_1\text{–}B_4, (RS), (ND^Z_{\equiv})\}$.

The steps in the procedure that leads to the completeness of $\{B_1\text{–}B_4, (RS), (ND^Z)\}$ can be adapted by substituting each reference to the observer I by T, thus obtaining a proof of the completeness of $\{B_1\text{–}B_4, (RS), (T\text{-}ND^Z_{\equiv})\}$ for $\leq_T^{l(Z)}$. However, the notion of head normal form for $N = I$ uses the fact that the summands $hnf^Z(q, a, i, X_1)$ can be defined in terms of the offers $X_1 \subseteq \mathcal{P}(Act)$, which correspond to the values produced by the observer I. For an arbitrary N, a more general definition of hnf's, valid for every observer, is needed.

Definition 5.10. For $p = \sum_{a \in X_0} \sum_{i \in I_a} a p^i_a$, its totally expanded Z-head normal form $tehnf^Z_N(p)$ is that given by:

- For $a \in X_0$, $i \in I_a$, and $K_a \subseteq I_a$ we consider a decomposition $p^k_a = p^{k_1}_a + p^{k_2}_a$ such that $M^N_Z(p^i_a, p^{k_1}_a, p^{k_2}_a)$. Then, $tehnf^Z_N(p, a, i, \langle (p^{k_1}_a, p^{k_2}_a)\rangle_{k \in K_a}) = a(p^i_a + \sum_{k \in K_a} p^{k_1}_a)$.
- $tehnf^Z_N(p) = \sum tehnf^Z_N(p, a, i, \langle (p^{k_1}_a, p^{k_2}_a)\rangle_{k \in K_a})$.

It is clear that for $K'_a \subseteq K_a$, or any decomposition $p^k_a = p^{k_3}_a + (p^{k_4}_a + p^{k_2}_a)$ with $p^{k_1}_a = p^{k_3}_a + p^{k_4}_a$, the corresponding $tehnf^Z_N(\ldots)$ is a subterm of $tehnf^Z_N(p, a, i, \langle p^{k_1}_a, p^{k_2}_a\rangle_{k \in K_a})$ and thus contributes nothing to the expanded normal form. This is the reason why we preferred the more compact definition of $hnf^Z(p)$ for semantics coarser than ready simulation.

Theorem 5.11. *For* $Z \in \{RT, FT, R, F\}$, $\{B_1\text{–}B_4, (TS), (T\text{-}ND^Z_{\equiv})\} \vdash p \preceq q$ *if and only if* $p \leq_T^{l(Z)} q$.

6. Logical characterization of semantics

The third and a very natural alternative to associate a semantics to processes lies in the logical framework. This is indeed quite a natural way to do it. We have a language to express properties of processes and a way to check whether a process satisfies a formula of the language: then, two processes are equivalent with respect to this semantics if and only if they satisfy the same set of formulas. In fact, the semantics can also be defined in terms of

Formulas \ Semantics ($\mathcal{Z}$)	T	S	CT	CS	F	FT	R	RT	PW	RS	PF	2S	B
$\top \in \mathcal{L}_\mathcal{Z}$	•	ν	•	ν	•	•	•	•	ν	ν	ν	ν	ν
$\mathbf{0} \in \mathcal{L}_\mathcal{Z}$			•	•	ν	ν	ν	ν	ν	ν	ν	ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z},\ a \in Act \Rightarrow$ $a\varphi \in \mathcal{L}_\mathcal{Z}$	•	•	•	•	•	•	•	•	ν	•	•	•	•
$X \subseteq Act \Rightarrow$ $\widetilde{X} \in \mathcal{L}_\mathcal{Z}$					•	ν	ν	ν	ν	ν	ν	ν	ν
$X \subseteq Act \Rightarrow$ $X \in \mathcal{L}_\mathcal{Z}$							•	ν	•	•	ν	ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z},\ X \subseteq Act \Rightarrow$ $\widetilde{X}\varphi \in \mathcal{L}_\mathcal{Z}$						•		ν	ν	ν		ν	ν
$\varphi \in \mathcal{L}_\mathcal{Z},\ X \subseteq Act \Rightarrow$ $X\varphi \in \mathcal{L}_\mathcal{Z}$								•	ν	ν		ν	ν
$\varphi_i \in \mathcal{L}_\mathcal{Z}\ \forall i \in I \Rightarrow$ $\bigwedge_{i\in I} \varphi_i \in \mathcal{L}_\mathcal{Z}$		•		•						•		•	•
$X \subseteq Act,\ \varphi_a \in \mathcal{L}_{PW}\ \forall a \in X \Rightarrow$ $\bigwedge_{a\in X} a\varphi_a \in \mathcal{L}_\mathcal{Z}$									•	ν		ν	ν
$\varphi_i, \varphi_j \in \mathcal{L}_T\ \forall i \in I \forall j \in J \Rightarrow$ $\bigwedge_{i\in I} \varphi_i \wedge \bigwedge_{i\in J} \neg\varphi_j \in \mathcal{L}_\mathcal{Z}$											•	ν	ν
$\varphi \in \mathcal{L}_S \Rightarrow$ $\neg\varphi \in \mathcal{L}_\mathcal{Z}$												•	ν
$\varphi \in \mathcal{L}_\mathcal{Z} \Rightarrow$ $\neg\varphi \in \mathcal{L}_\mathcal{Z}$													•

Table 3: Van Glabbeek's logical characterizations for the semantics in the ltbt spectrum.

the induced preorder that indicates whether a process satisfies more formulas than another one.

Each subset $\mathcal{L}$ of $\mathcal{L}_{HM}$ induces a semantics as stated in the following definition.

Definition 6.1. Any subset $\mathcal{L}$ of $\mathcal{L}_{HM}$ induces a logical semantics for processes, given by the preorder $\sqsubseteq_\mathcal{L}$: $p \sqsubseteq_\mathcal{L} q$ whenever for all $\varphi \in \mathcal{L}$, if $p \models \varphi$ then $q \models \varphi$. We say that $\mathcal{L}$ and $\mathcal{L}'$ are equivalent, and we write $\mathcal{L} \sim \mathcal{L}'$, if they induce the same semantics, that is $\sqsubseteq_\mathcal{L} = \sqsubseteq_{\mathcal{L}'}$.

Let us start with a look at Table 3, which contains the logical characterization of each of the semantics in van Glabbeek's spectrum. $\mathcal{L}_Z$ with $Z \in \{T, CT, F, FT, R, RT, PF, S, CS, RS, 2S, PW, B\}$ denotes each of the logics; the dots indicate the clauses needed to obtain the corresponding languages; and the boxes marked with ν correspond to rules that could be added to $\mathcal{L}_Z$, but would only introduce redundant formulas. The following constructs, which appear in the table but are not in $\mathcal{L}_{HM}$, can be obtained as syntactic sugar:

$$\widetilde{X} := \bigwedge_{a\in X} \neg a\top \qquad \widetilde{X}\varphi' := \widetilde{X} \wedge \varphi' \qquad 0 := \widetilde{Act}$$

$$\varphi_1 \wedge \varphi_2 := \bigwedge_{i\in\{1,2\}} \varphi_i \qquad X := \bigwedge_{a\in X} a\top \wedge \bigwedge_{a\notin X} \neg a\top \qquad X\varphi' := X \wedge \varphi' \qquad \widetilde{a} := \neg a\top$$

Disjunction does not appear in $\mathcal{L}_{HM}$ and therefore neither in any of the logics $\mathcal{L}_Z$ characterizing the semantics in the ltbt spectrum. It is probably folklore that it can be added in all cases without affecting the expressive power of each of these logics, but since we have not found a clear statement in this direction in any of our references, next we establish the result and comment on its proof.

Proposition 6.2. *Let us define $\mathcal{L}_Z^\vee$, with $Z \in \{T, CT, F, FT, R, RT, PF, S, CS, RS, 2S, PW, B\}$, by adding the clause $\bigvee_{i\in I} \varphi_i \in \mathcal{L}_Z^\vee$ if $\varphi_i \in \mathcal{L}_Z^\vee$ for all $i \in I$ to the clauses that define $\mathcal{L}_Z$,*

replacing $\mathcal{L}_Z$ *by* $\mathcal{L}_Z^\vee$ *in the other clauses, and making* $p \models \bigvee \varphi_i$ *iff there exists* $i \in I$ *with* $p \models \sigma_i$. *Then,* $\mathcal{L}_Z^\vee \sim \mathcal{L}_Z$.

Proof. It is interesting to observe that even if the result is valid for all the semantics, the reason behind is not the same as for bisimulation. In that case, we only need to apply the De Morgan laws to get the "definition" of $\vee$ as a combination of $\neg$ and $\wedge$. However, for the rest of the semantics we do not have negation as "constructor", but $\vee$ distributes over $\wedge$ and the prefix operator (that is $\bigvee a\varphi_i = a \bigvee \varphi_i$), while negation is never applied to a formula $\varphi' \in \mathcal{L}_Z^\vee$. Therefore, by floating to the top any $\vee$, using those distribution laws, a formula in $\mathcal{L}_Z^\vee$ becomes equivalent to a disjunction of formulas within the corresponding language $\mathcal{L}_Z$, and the equivalence of both logics follows. □

As we will see in this section, each of our logics is defined by a set of rules and, as usual, only the formulas that can be obtained by finite application of these rules belong to the logics. One important feature of our approach is that instead of focusing on small sets of formulas characterizing each of the semantics, we somewhat follow the opposite approach by including all the formulas, from a certain family, that are preserved by each of the semantics. This choice has many interesting side effects. In particular, we will not need to look for adequate formulas reflecting the characteristics of each of the semantics, but instead pick up from our "repository" of possible formulas those that are preserved by the current semantics. Thus, we characterize each of the semantics by means of the formulas that "see" the kind of observations that define it. As a consequence, we know whether a semantics is coarser than another by checking whether the logic characterizing the former is included in the logic characterizing the latter. Moreover, by using a larger logic we may find a formula expressing some property that is preserved by the corresponding semantics, while if we settle on a smaller logic we might need a collection of formulas to express a simple property.

Formally speaking, for each semantics defined by a preorder $\prec$ we have a language $\mathcal{L} \subseteq \mathcal{L}_{HM}$ characterizing it: $\varphi \in \mathcal{L}$ iff $((p \prec q \wedge p \models \varphi)$ implies $q \models \varphi)$. However, it is not easy (nor specially illustrative) to capture the whole set of formulas characterizing the semantics. Instead, we will consider sufficiently large families defined in a simple way that provide natural characterizations of the different semantics and show the relationship between them so that, as stated above, whenever a semantics is finer than another, the logic characterizing the first will contain that for the latter.

As will become clear when we introduce our new logical characterizations, Table 3 readily presents the features that allow us to classify the semantics in the spectrum in four categories:

- Bisimulation semantics, characterized by HML, that is closed under negation ($\neg$), so that the preorder defined is an equivalence (the bisimulation). The remaining semantics are defined by non-trivial preorders, i.e., the preorders are not equivalences and their logical characterizations are, of course, not closed under negation.
- Simulation semantics (S, CS, RS, ...), characterized by branching observations, which will be reflected by the unrestricted use of the operator $\bigwedge$ in the formulas.
- Linear semantics (T, F, R, ...), characterized by linear observations. We will get them by severely restricting the use of $\bigwedge$ and the use of the negation.
- Deterministic branching semantics, corresponding to an intermediate class between branching and linear semantics, where determinism appears restricting the use of the operator $\bigwedge$

in combination with the prefix operator. The only semantics in this class in the classical spectrum is PW.

As already happened in Sections 3 and 4, our unified logical semantics will provide an enlarged spectrum—Figure 11. In particular, we will show the logical characterization of revivals semantics, introduced by Roscoe in [48] and already axiomatized in [19].

6.1. **A new logical characterization of the most popular semantics.** Again, we start with the best known classical semantics, that is, those at the layer of ready simulation in the spectrum. All of them use in some way the set of formulas $\mathcal{L}_I = \{a\top \mid a \in Act\}$ that characterizes the initial offers of a process. In Section 6.2 we will present the logics for the rest of the semantics in a unified way, remarking how they are obtained similarly to those in this section but working from the set $\mathcal{L}_N$ of formulas associated to the corresponding constraint N.

We will prove the equivalence between each of our logics and the corresponding logical characterization defined by van Glabbeek, thus checking that our new logical characterizations are indeed correct. But one of the intended goals of our unification was to obtain direct and natural proofs. This will be illustrated in Section 7 by showing the equivalence between each of our logical semantics and the corresponding observational semantics of Section 4. This will provide a new, single proof of their correctness without having to resort to the characterizations defined by van Glabbeek.

Definition 6.3. *Ready simulation semantics.* We define the set of formulas $\mathcal{L}'_{RS}$ for ready simulation semantics by:

- If $\sigma \in \mathcal{L}_I$ then $\sigma \in \mathcal{L}'_{RS}$;
- if $\sigma \in \mathcal{L}_I$ then $\neg\sigma \in \mathcal{L}'_{RS}$;
- if $\varphi_i \in \mathcal{L}'_{RS}$ for all $i \in I$ then $\bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{RS}$;
- if $\varphi \in \mathcal{L}'_{RS}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{RS}$.

Ready trace semantics. We define the set of formulas $\mathcal{L}'_{RT}$ for ready trace semantics by:

- $\top \in \mathcal{L}'_{RT}$;
- if $\varphi \in \mathcal{L}'_{RT}$ and $X_1, X_2 \subseteq \mathcal{L}'_I$ then $(\bigwedge_{a\in X_1} a\top \wedge \bigwedge_{b\in X_2} \neg b\top) \wedge \varphi \in \mathcal{L}'_{RT}$;
- if $\varphi \in \mathcal{L}'_{RT}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{RT}$.

Failure trace semantics. We define the set of formulas $\mathcal{L}'_{FT}$ for failure trace semantics by:

- $\top \in \mathcal{L}'_{FT}$;
- if $\varphi \in \mathcal{L}'_{FT}$ and $X_1 \subseteq \mathcal{L}'_I$ then $(\bigwedge_{a\in X_1} \neg a\top) \wedge \varphi \in \mathcal{L}'_{FT}$;
- if $\varphi \in \mathcal{L}'_{FT}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{FT}$.

Readiness semantics. We define the set of formulas $\mathcal{L}'_R$ for readiness semantics by:

- $\top \in \mathcal{L}'_R$;
- if $X_1 \subseteq \mathcal{L}'_I$ and $X_2 \subseteq \mathcal{L}'_I$ then $(\bigwedge_{a\in X_1} a\top \wedge \bigwedge_{b\in X_2} \neg b\top) \in \mathcal{L}'_R$;
- if $\varphi \in \mathcal{L}'_R$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_R$.

Failures semantics. We define the set of formulas $\mathcal{L}'_F$ for failures semantics by:

- $\top \in \mathcal{L}'_F$;
- if $X_1 \subseteq \mathcal{L}'_I$ then $(\bigwedge_{a\in X_1} \neg a\top) \in \mathcal{L}'_F$;
- if $\varphi \in \mathcal{L}'_F$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_F$.

It is immediate that $\mathcal{L}'_{RS} \subseteq \mathcal{L}_B$ and hence ready simulation semantics is coarser than bisimulation equivalence. We also have $\mathcal{L}'_F \subseteq \mathcal{L}'_R$, $\mathcal{L}'_F \subseteq \mathcal{L}'_{FT}$, $\mathcal{L}'_R \subseteq \mathcal{L}'_{RT}$, $\mathcal{L}'_{FT} \subseteq \mathcal{L}'_{RT}$, and $\mathcal{L}'_{RT} \subseteq \mathcal{L}'_{RS}$, which can be interpreted in a similar way. Let us now focus our attention on the third rule of the definition of $\mathcal{L}'_{RS}$: the unrestricted use of conjunction corresponds to the branching nature of the semantics. Moreover, the two first rules allow to fix the set of offers of a process as I-simulations impose. By contrast, the linear semantics only allow the use of conjunction to join those simple formulas that fix the set of offers along a computation (in the case of the readies-based semantics), or their over-approximations (obtained by means of the negated formulas $\neg a\top$, in the case of the failures-based semantics). Finally, notice how these simple formulas can only be checked at the corresponding final state, for the two simpler coarser semantics.

Now, for $Z \in \{RS, RT, FT, R, F\}$, each of the logics $\mathcal{L}'_Z$ is a superset of the corresponding logic $\mathcal{L}_Z$ defined in Table 3. To be precise, for FT and F we need to remove the syntactic sugar used by van Glabbeek as stated below.

Remark 6.4. We have used in Section 4.2 X^c to denote the complementary of a set, because previously in Definition 4.20 we used the classic over line notation to refer to closures of sets $\mathcal{T} \subseteq LGO_N$. However, since we will not need those closure operators anymore we prefer to used the classic notation referring the complement of a set X by $\overline{X}$.

Proposition 6.5.
(1) $\mathcal{L}_{RS} \subsetneq \mathcal{L}'_{RS}$.
(2) $\mathcal{L}_{RT} \subsetneq \mathcal{L}'_{RT}$.
(3) $\mathcal{L}'_{FT} \supseteq$ *desugared*$(\mathcal{L}_{FT})$, *where the desugaring function removes the syntactic sugar used in* $\mathcal{L}_{FT}$.
(4) $\mathcal{L}_R \subsetneq \mathcal{L}'_R$.
(5) $\mathcal{L}'_F \supseteq$ *desugared*$(\mathcal{L}_F)$, *where the desugaring function removes the syntactic sugar used in* $\mathcal{L}_F$.

Proof. Recall the definition of $\mathcal{L}_Z$ in Table 3.

(1) To prove that $\mathcal{L}_{RS} \subseteq \mathcal{L}'_{RS}$, it is sufficient to show that each formula $\varphi_X = \bigwedge_{a\in X} a\top \wedge \bigwedge_{b\notin X} \neg b\top$ corresponding to $X \subseteq Act$ belongs to $\mathcal{L}'_{RS}$. Both $a\top$ and $\neg b\top$ are in $\mathcal{L}'_{RS}$ and the combination of these formulas with the operator $\wedge$ is also in the set $\mathcal{L}'_{RS}$. For the inclusion to be proper, it is sufficient to notice that the formula $\neg b\top$ belongs to $\mathcal{L}'_{RS}$ but not to the set $\mathcal{L}_{RS}$.
(2) To prove that $\mathcal{L}_{RT} \subseteq \mathcal{L}'_{RT}$ it is sufficient to show that for every $X \subseteq Act$ and any $\varphi \in \mathcal{L}_{RT}$, the formula $(\bigwedge_{a\in X} a\top \wedge \bigwedge_{b\notin X} \neg b\top) \wedge \varphi$ belongs to $\mathcal{L}'_{RT}$. Note that $b \notin X$ is equivalent to $b \in \overline{X}$, so taking $X_1 = X$ and $X_2 = \overline{X}$ we have that the considered formula belongs to $\mathcal{L}'_{RT}$. To prove that $\mathcal{L}_{RT} \subset \mathcal{L}'_{RT}$, it is sufficient to note that $(\neg b\top)\wedge\varphi$ belongs to $\mathcal{L}'_{RT}$, by taking $X_1 = \emptyset$ and $X_2 = \{b\}$, but it does not belong to $\mathcal{L}_{RS}$.
(3) In this case the result is trivial, since the definitions of $\mathcal{L}_{FT}$ and $\mathcal{L}'_{FT}$ are almost the same, once the syntactic sugar is removed. The only difference is that $\bot \in \mathcal{L}'_{FT}$, which obviously does not affect the inclusion.
(4) To prove that $\mathcal{L}_R \subseteq \mathcal{L}'_R$, it is sufficient to show that for every $X \subseteq Act$ the formula $\bigwedge_{a\in X} a\top \wedge \bigwedge_{b\notin X} \neg b\top$ belongs to $\mathcal{L}'_R$. Note that the condition $b \notin X$ is equivalent to $b \in \overline{X}$, so taking $X_1 = X$ and $X_2 = \overline{X}$ we have that the considered formula belongs to $\mathcal{L}'_R$. To check that $\mathcal{L}_R \subsetneq \mathcal{L}'_R$, it is sufficient to note that the formula $\neg b\top$ belongs to $\mathcal{L}'_R$ by taking $X_1 = \emptyset$ and $X_2 = \{b\}$, while it does not belong to $\mathcal{L}_R$.

(5) Analogous to 3. □

As stated earlier, in order to obtain more natural characterizations, our logics typically contain large sets of formulas. This is why in most cases our logics contain those proposed by van Glabbeek. In order to prove the equivalence between ours and his, we have to show that our additional formulas are in fact redundant and could be safely removed.

Proposition 6.6. *(1) $\mathcal{L}_{RS} \sim \mathcal{L}'_{RS}$; (2) $\mathcal{L}_{RT} \sim \mathcal{L}'_{RT}$; (3) $\mathcal{L}_{FT} \sim \mathcal{L}'_{FT}$; (4) $\mathcal{L}_{R} \sim \mathcal{L}'_{R}$; and (5) $\mathcal{L}_{F} \sim \mathcal{L}'_{F}$.*

Proof.

(1) Any conjunction and negation of formulas in $\mathcal{L}_I$ can be obtained as the disjunction of the formulas X describing all the "compatible" offers. These are those including the positive and negative information in the corresponding conjunction, i.e., $a\top \sim \bigvee_{a\in X} X$; $\neg a\top \sim \bigvee_{a\notin X} X$. Then, by applying Proposition 6.2, we obtain $\mathcal{L}'_{RS} \sim \mathcal{L}_{RS}$.
(2) We have shown that the formulas in $\mathcal{L}_{RT}$ are particular cases of the formulas in $\mathcal{L}'_{RT}$: those that completely define the offers at the states along a computation (when we apply the second clause in the definition of $\mathcal{L}'_{RT}$ with $X_2 = \overline{X_1}$). In contrast, our more general formulas $(\bigwedge_{a\top\in X_1} a\top \wedge \bigwedge_{b\top\in X_2} \neg b\top)\wedge\varphi$, where $\varphi \in \mathcal{L}'_{RT}$, could provide us with some partial information, combining both positive information $a\top \in X_1$ and negative information $b\top \in X_2$, which tells us that we are in an arbitrary state X satisfying $X_1 \subseteq X \subseteq \overline{X_2}$. But we can replace these formulas by the disjunction of all the formulas describing any of these possible offers X. By repeating this procedure at each level of the formula, we finally obtain a disjunction of formulas in $\mathcal{L}_{RT}$. To conclude, it is enough to apply Proposition 6.2.
(3) We know $\bot = \neg\top = \bigvee_{i\in\emptyset} \varphi_i$, and applying Proposition 6.2 we get the equivalence.
(4) Note that van Glabbeek allowed in $\mathcal{L}_R$ only "normal form" formulas from $\mathcal{L}'_R$, which can give us information about the offers at the final state in a computation (when we apply the second clause in the definition of $\mathcal{L}'_R$) or simply define these computations by means of the prefix operator (when we apply the third clause in the definition of $\mathcal{L}'_R$). However, our more general formulas $(\bigwedge_{a\top\in X_1} a\top \wedge \bigwedge_{b\top\in X_2} \neg b\top)$ can also provide us with some partial information about the final state, which could be both positive $a\top \in X_1$ and negative $b\top \in X_2$. In the (allowed) case $X_1 \bigcap X_2 \neq \emptyset$ we have that the formula is unsatisfiable. Otherwise, we are offering the actions a corresponding to formulas $a\top$ in any $X \subseteq \mathcal{L}_I$ that satisfies $X_1 \subseteq X$ and $X \subseteq \overline{X_2}$, and we can replace again the corresponding formula by a disjunction of formulas in $\mathcal{L}_R$.
(5) Analogous to 3. □

In the following, when we consider a logic $\mathcal{L}_Z$ and the index Z refers to some concrete semantics, as is the case with RS, RT, FT, R, and F above, by abuse of notation we will simply write $\sqsubseteq'_Z$ instead of $\sqsubseteq_{\mathcal{L}'_Z}$ for the preorder induced by the logic $\mathcal{L}'_Z$.

Theorem 6.7.

(1) *The logical semantics $\sqsubseteq'_{RS}$ induced by the logic $\mathcal{L}'_{RS}$ is equivalent to the observational branching semantics defined by $\leq^b_I$, generated by the set of branching general observations BGO_I.*
(2) *For $Z \in \{F, FT, R, RT\}$, the logical semantics $\sqsubseteq'_Z$ induced by the logic $\mathcal{L}'_Z$ is equivalent to the observational linear semantics $\leq^{l(Z)}_I$ in Definitions 4.14 and 4.19.*

Proof. It is a consequence of Proposition 6.6 and the results by van Glabbeek collected in Table 3, Theorem 4.9, and Proposition 4.18.

(1) We have already checked that our formulas are equivalent to van Glabbeek's: $\mathcal{L}'_{RS} \sim \mathcal{L}_{RS}$. It is easy to show that once we have eliminated the unsatisfiable formulas in $\mathcal{L}'_{RS}$ (those that simultaneously make two different offers, or perform an action that was not included in the corresponding offer) the remaining formulas in $\mathcal{L}'_{RS}$ admit a normal form in the language $\mathcal{N}(\mathcal{L}_{\mathcal{RS}})$, which we define as follows:
- if $X \subseteq Act$, $\{a_i \mid i \in I\} \subseteq X$, and $\varphi_i \in \mathcal{N}(\mathcal{L}_{\mathcal{RS}})$, then $(\bigwedge_{b\in X} b\top \wedge \bigwedge_{b\notin X} \neg b\top) \wedge \bigwedge_{i\in I} a_i\varphi_i \in \mathcal{N}(\mathcal{L}_{\mathcal{RS}})$;
- if $\{a_i \mid i \in I\} \subseteq Act$ and $\varphi_i \in \mathcal{N}(\mathcal{L}_{\mathcal{RS}})$ then $\bigwedge_{i\in I} a_i\varphi_i \in \mathcal{N}(\mathcal{L}_{\mathcal{RS}})$.

Within this set, consider the subset of formulas $\mathcal{CN}(\mathcal{L}_{\mathcal{RS}})$ which can be generated using the first clause in the above definition. We can establish an isomorphism between $\mathcal{CN}(\mathcal{L}_{\mathcal{RS}})$ and the set of possible branching general observations BGO_I. Moreover, it is easy to prove that if for every formula $\varphi \in \mathcal{CN}(\mathcal{L}_{\mathcal{RS}})$ we define bgo_φ as the corresponding observation, then $\varphi \models p$ iff $bgo_\varphi \in BGO_I(p)$, from which it immediately follows that $\mathcal{CN}(\mathcal{L}_{\mathcal{RS}})$ characterizes the ready simulation semantics defined via BGO_I.

Now, to conclude the proof it is sufficient to show that $\mathcal{N}(\mathcal{L}_{\mathcal{RS}})$ and $\mathcal{CN}(\mathcal{L}_{\mathcal{RS}})$ are equivalent. Note that whenever we use the second clause in the definition of $\mathcal{N}(\mathcal{L}_{\mathcal{RS}})$, we are ignoring the possibility of specifying the offer X at the state we are. As a consequence, the offer could be any satisfying $\{a_i \mid i \in I\} \subseteq X$, for the corresponding set $\{a_i \mid i \in I\}$. Then we can complete the associated formula $\bigwedge_{i\in I} a_i\varphi_i$ by adding the disjunction $\bigvee_{\{a_i/i\in I\}\subseteq X}(\bigwedge_{b\in X} b\top \wedge \bigwedge_{b\notin X} \neg b\top)$. Floating all the disjunctions away we obtain a disjunction of formulas in $\mathcal{N}(\mathcal{L}_{\mathcal{RS}})$, which ends the proof.

(2) • If $Z = RT$, we know that $\mathcal{L}'_{RT} \sim \mathcal{L}_{RT}$. It is easy to show that eliminating all the unsatisfiable formulas (those that simultaneously offer two different sets of actions, or perform an action a that is not included in the corresponding offer X) the rest of the formulas in $\mathcal{L}'_{RT}$ admit a normal form in the language $\mathcal{N}(\mathcal{L}_{\mathcal{RT}})$, which we define as follows:
  – if $X \subseteq Act$ then $(\bigwedge_{b\in X} b\top \wedge \bigwedge_{b\notin X} \neg b\top) \in \mathcal{N}(\mathcal{L}_{\mathcal{RT}})$;
  – if $X \subseteq Act$, $a \in X$, and $\varphi \in \mathcal{N}(\mathcal{L}_{\mathcal{RT}})$ then $(\bigwedge_{b\in X} b\top \wedge \bigwedge_{b\notin X} \neg b\top) \wedge a\varphi \in \mathcal{N}(\mathcal{L}_{\mathcal{RT}})$;
  – $\top \in \mathcal{N}(\mathcal{L}_{\mathcal{RT}})$;
  – if $a \in Act$ and $\varphi \in \mathcal{N}(\mathcal{L}_{\mathcal{RT}})$ then $a\varphi \in \mathcal{N}(\mathcal{L}_{\mathcal{RT}})$.

  As we did for the case of ready simulation, we could define the corresponding language of complete formulas $\mathcal{CN}(\mathcal{L}_{\mathcal{RT}})$. The formulas in $\mathcal{L}'_{RT}$ that we obtained in the proof of Proposition 6.6, for the case of RT, are indeed in $\mathcal{CN}(\mathcal{L}_{\mathcal{RT}})$ because any subformula gives us some partial information about the offers at the corresponding state, which in the worst case could be empty. Therefore, when we translate this information into the language $\mathcal{L}'_{RT}$ we obtain a disjunction between complete formulas in $\mathcal{CN}(\mathcal{L}_{\mathcal{RT}})$. We can easily establish the isomorphism between $\mathcal{CN}(\mathcal{L}_{\mathcal{RT}})$ and the domain LGO_I, and then prove that for every formula $\varphi \in \mathcal{CN}(\mathcal{L}_{\mathcal{RT}})$, if we define lgo_φ as the corresponding observation, we have $\varphi \models p$ iff $lgo_\varphi \in LGO_I(p)$. From here it follows that $\mathcal{CN}(\mathcal{L}_{\mathcal{RT}})$ characterizes the ready simulation semantics defined via LGO_I. To conclude the proof we need to show that $\mathcal{N}(\mathcal{L}_{\mathcal{RT}})$ and $\mathcal{CN}(\mathcal{L}_{\mathcal{RT}})$ are equivalent, which is analogous to $\mathcal{N}(\mathcal{L}_{\mathcal{RS}})$ and $\mathcal{CN}(\mathcal{L}_{\mathcal{RS}})$ above.

• $Z = FT$. ($\Rightarrow$) Let p and q be such that $p \sqsubseteq'_{FT} q$: we will show that $p \leq_I^{l\supseteq} q$. Given an observation $X_0a_1X_1 \dots a_nX_n \in LGO_I(p)$, we have a failure trace $\overline{X_0}a_1\overline{X_1} \dots a_n\overline{X_n}$

for the process p. Now, we consider the formulas $\varphi_n = \bigwedge_{a\in\overline{X}} \neg a\top$, $\varphi_i = \bigwedge_{a\in\overline{X_i}} \neg a\top \wedge a_{i+1}\varphi_{i+1}$ with $i \in 0..n-1$, and we have that $p \models \varphi_0$. Therefore $q \models \varphi_0$, which means that $\overline{X_0}a_1\overline{X_1}\dots a_n\overline{X_n}$ is a failure trace of q. Then, there is some $Y_0a_1Y_2\dots a_nY_n \in LGO_I(p)$ with $Y_i \bigcap \overline{X_i} = \emptyset$ for all $i = 0..n$ or, equivalently, $X_i \supseteq Y_i$ for all $i = 0..n$. As a result, $LGO_I(p) \leq_I^{l\supseteq} LGO_I(q)$, which means $p \leq_I^{l\supseteq} q$.
($\Leftarrow$) Let us suppose that for all $X_0a_1X_1\dots a_nX_n \in LGO_I(p)$ there exists $Y_0a_1Y_1\dots a_nY_n \in LGO_I(q)$ such that $X_i \supseteq Y_i$ for all $i = 0..n$; we want to show that if $p \models \varphi$ then $q \models \varphi$, for all $\varphi \in \mathcal{L}'_{FT}$. If $p \models \varphi$, we can decompose φ by means of a sequence of formulas, taking $\varphi = \varphi_n$, $\varphi_i = \bigwedge_{a\in X_2^i} \neg a\top \wedge a_i\varphi_{i-1}$ for $i \in 1..n$ and $\varphi_0 = \bigwedge_{a\in X_2^0} \neg a\top$. Therefore, $X_na_nX_{n-1}\dots a_1X_0$ is a failure trace for the process p, so there exists $Z_na_nZ_{n-1}\dots a_1Z_0 \in LGO_I(p)$ with $Z_i \bigcap X_i = \emptyset$, and using that $p \leq_I^{l\supseteq} q$, there exists some $Y_na_nY_{n-1}\dots a_1Y_0 \in LGO_I(q)$ with $Y_i \subseteq Z_i$, so that $Y_i \bigcap X_i = \emptyset$ and then we get $q \models \varphi_n$.

- If $Z = R$, using the result in the proof of Proposition 6.6 for the case of R it is enough to show the result for the set of "normal form" formulas $\mathcal{N}(\mathcal{L}_\mathcal{R})$ defined by:
  – if $X \subseteq Act$ then $(\bigwedge_{b\in X} b\top \wedge \bigwedge_{b\notin X} \neg b\top) \in \mathcal{N}(\mathcal{L}_\mathcal{R})$;
  – $\top \in \mathcal{N}(\mathcal{L}_\mathcal{R})$;
  – $a \in Act$ and $\varphi \in \mathcal{N}(\mathcal{L}_\mathcal{R})$ then $\varphi \in \mathcal{N}(\mathcal{L}_\mathcal{R})$.
  ($\Rightarrow$) Let p and q be such that $p \sqsubseteq'_R q$: we will show $p \leq_I^{lf} q$. Given an observation $X_0a_1X_1\dots a_nX_n \in LGO_I(p)$, it corresponds to the readiness information $a_1\dots a_nX_n$ of p. Now, we consider the formulas $\varphi_n = \bigwedge_{a\in X} a\top \wedge \bigwedge_{a\notin X} \neg a\top$; $\varphi_{i-1} = a_i\varphi_i$ with $i \in 1\dots n-1$, and we have that $p \models \varphi_0$. Therefore $q \models \varphi_0$, and $a_1\dots a_nX_n$ is a readiness information of q and, as a consequence, there is an observation $Y_0a_1Y_2\dots a_nY_n \in LGO_I(q)$ with $Y_n = X_n$, proving $p \leq_I^{lf} q$.
  ($\Leftarrow$) Let us suppose that for all $X_0a_1X_1\dots a_nX_n \in LGO_I(p)$ there exists some $Y_0a_1Y_1\dots a_nY_n \in LGO_I(q)$ such that $X_n = Y_n$. We want to show that if $p \models \varphi$ then $q \models \varphi$ for all $\varphi \in \mathcal{CN}(\mathcal{L}_\mathcal{R})$. If $p \models \varphi$, we can decompose φ taking $\varphi = \varphi_n$, $\varphi_i = a_i\varphi_{i-1}$, for all $i \in 1..n$, and $\varphi_0 = \bigwedge_{a\in X_0} a\top \wedge \bigwedge_{a\notin X_0} \neg a\top$. Then we have that $a_na_{n-1}\dots a_1X_0$ is a readiness information of p, so there exists some $Z_na_nZ_{n-1}\dots a_1X_0 \in LGO_I(p)$, and some $Y_na_nY_{n-1}\dots a_1Y_0 \in LGO_I(q)$ with $Y_0 = X_0$, from which we conclude that $q \models \varphi_n$.
- $Z = F$. ($\Rightarrow$) Let p and q be such that $p \sqsubseteq'_F q$: we will show $p \leq_I^{lf\supseteq} q$. Given an observation $X_0a_1X_1\dots a_nX_n \in LGO_I(p)$, it generates a (maximal) failure $a_1\dots a_n\overline{X_n}$ of the process p. Now, we consider the formulas $\varphi_n = \bigwedge_{a\in\overline{X}} \neg a\top$; $\varphi_{i+1} = a_{i+1}\varphi_i$ with $i \in 0..n-1$, and we have that $p \models \varphi_0$. Therefore, $q \models \varphi_0$, so $a_1\dots a_n\overline{X_n}$ is a failure information of q, and there is some $Y_0a_1Y_2\dots a_nY_n \in LGO_I(q)$ with $Y_n \bigcap \overline{X_n} = \emptyset$, or equivalently $X_n \supseteq Y_n$, proving that $p \leq_I^{lf\supseteq} q$.
  ($\Leftarrow$) Let us suppose that for all $X_0a_1X_1\dots a_nX_n \in LGO_I(p)$ there exists some $Y_0a_1Y_1\dots a_nY_n \in LGO_I(q)$ such that $X_n \supseteq Y_n$. We want to show that if $p \models \varphi$ then $q \models \varphi$ for all $\varphi \in \mathcal{L}'_F$. If $p \models \varphi$, we can decompose φ taking $\varphi = \varphi_n$, $\varphi_i = a_i\varphi_{i-1}$, with $i \in 1..n$, and $\varphi_0 = \bigwedge_{a\in X_0} \neg a\top$. From $p \models \varphi$ we infer that $a_na_{n-1}\dots a_1X_0$ is a failure information of the process p, so there exists $Z_na_nZ_{n-1}\dots a_1Z_0 \in LGO_I(p)$ with $Z_0 \bigcap X_0 = \emptyset$, and then there is some $Y_na_nY_{n-1}\dots a_1Y_0 \in LGO_I(q)$ with $Y_n \subseteq Z_n$, so that $Y_n \bigcap X_n = \emptyset$, obtaining $q \models \varphi_n$.

□

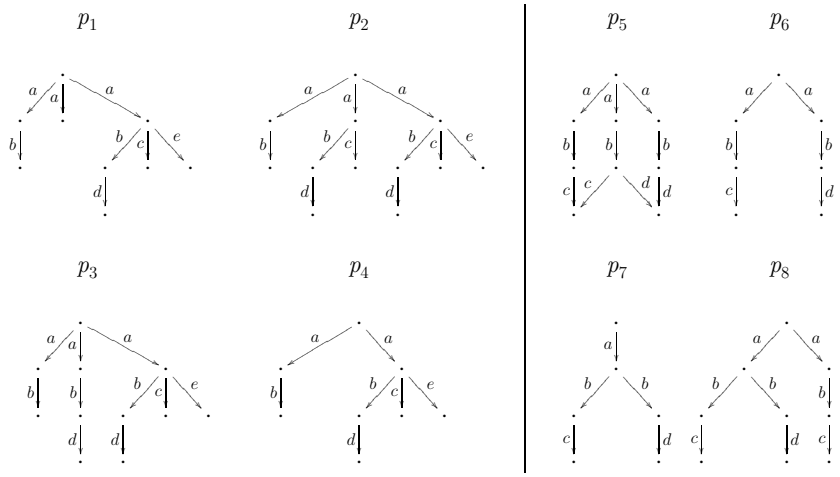


Figure 13: A simple example to show the strength of the different logics

Example 6.8. Figure 13 shows a collection of examples to illustrate the differences between the semantics in the *RS* layer of the spectrum. All the following equivalences can be checked by taking any arbitrary formula from the logic defining each of the semantics. For readability, we omit the last $\top$ in all subformulas. Besides, $\sim_X$ (resp. $\nsim_X$), where X is a set of indexes, represents any $\sim_Z$ (resp. $\nsim_Z$), with $Z \in X$.

- $p_1 \not\sqsubseteq'_F p_2$ and $p_1 \not\sqsubseteq'_{\{R,\, FT,\, RT,\, RS\}} p_2$ because $p_1 \models a(\neg b \wedge \neg c)$, but p_2 does not satisfy it.
- $p_2 \sim_F p_3$, but $p_2 \not\sqsubseteq'_{\{R,\, FT\}} p_3$ and thus $p_2 \not\sqsubseteq'_{\{RT,\, RS\}} p_3$, since p_2 satisfies $a(\neg e \wedge c)$ but p_3 does not.
- $p_3 \sim_{\{F,\, R\}} p_4$, but $p_3 \not\sqsubseteq'_{FT} p_4$ and thus $p_3 \not\sqsubseteq'_{\{RT,\, RS\}} p_4$, because p_3 satisfies $a(\neg c \wedge b(\neg e \wedge d))$ but p_4 does not.
- $p_5 \sim_{\{F,\, FT\}} p_6$, but $p_5 \not\sqsubseteq'_R p_6$ and thus $p_5 \not\sqsubseteq'_{\{RT,\, RS\}} p_6$, since p_5 satisfies $ab(c \wedge d)$ but p_6 does not.
- $p_6 \sim_{\{F,\, R,\, RT,\, FT\}} p_7$ but $p_7 \not\sqsubseteq'_{RS} p_6$, because p_7 satisfies $a(bc \wedge bd)$ but p_6 does not.
- $p_7 \sim_{\{F,\, R,\, RT,\, FT,\, RS\}} p_8$.

6.2. **Our new unified logical characterizations of the semantics.** Inspired by the semantics studied in Section 6.1, next we define the general format for the logics characterizing each of the semantics in the spectrum. We start by enlarging the spectrum yet a bit more.

Definition 6.9.

(1) *Universal semantics.* We define the set $\mathcal{L}'_U$ of universal formulas that characterize the trivial semantics that identifies all the processes by $\mathcal{L}'_U = \{\top\}$.

(2) *Complete semantics.* We define the set $\mathcal{L}'_C$ of complete formulas characterizing the semantics that only distinguishes the terminated processes from the non-terminated ones by $\mathcal{L}'_C = \{\top, \neg 0\}$.

(3) *Initial offer semantics*. We define the set $\mathcal{L}'_I$ of initial offer formulas characterizing the semantics that only observers the set of initial actions of a process by $\mathcal{L}'_I = \{\top, \neg 0\} \cup \{a\top \mid a \in Act\}$.

In the definition above the subformula $\neg 0$ is just syntactic sugar for the formula $\neg(\bigwedge_{a\in Act} \neg a\top)$. Therefore, once again all these new logics are sublogics of $\mathcal{L}_{HM}$ and, as a result, we do not need to define their semantics.

Note that $\mathcal{L}'_I$ is a bit larger than the logic $\mathcal{L}_I$ from Section 6.1. Once again, this is so in order to get a more uniform presentation of our logics: $\neg 0$ is indeed redundant. By including it we immediately obtain that the complete semantics is coarser than the initial offer semantics, because $\mathcal{L}'_C \subseteq \mathcal{L}'_I$. Based on this result we will also obtain that the complete simulation is coarser than the ready simulation. Certainly, $\neg 0$ is redundant in $\mathcal{L}'_I$ (but not in $\mathcal{L}'_C$!), because by means of it we can only distinguish a process that cannot execute any action from any other that can execute someone. But using the corresponding $a\top$ formula we can also get that.

6.2.1. *The simulation semantics.* As repeatedly noted, the family of simulation semantics constitute the spine of the new spectrum. All of them are defined in a homogeneous way thanks to the notion of constrained simulation from [24]. Next we present their logical characterization.

Definition 6.10. Given a set of formulas $\mathcal{L}'_N$ defining a semantics N, we define the set of formulas $\mathcal{L}'_{NS}$ that characterizes the N-constrained simulation semantics by:

- If $\sigma \in \mathcal{L}'_N$ then $\sigma \in \mathcal{L}'_{NS}$;
- if $\sigma \in \mathcal{L}'_N$ then $\neg\sigma \in \mathcal{L}'_{NS}$;
- if $\varphi_i \in \mathcal{L}'_{NS}$ for all $i \in I$ then $\bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{NS}$;
- if $\varphi \in \mathcal{L}'_{NS}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{NS}$.

Taking $N \in \{U, C, I\}$ we obtain $\mathcal{L}'_{US}$, $\mathcal{L}'_{CS}$ and $\mathcal{L}'_{IS}$, that we rewrite as $\mathcal{L}'_S$ and $\mathcal{L}'_{RS}$ in the first and last cases to emphasize the classic notation for simulation semantics. From $\mathcal{L}'_S$ we obtain $\mathcal{L}'_{SS}$, that we will denote as $\mathcal{L}'_{2S}$. To complete the collection of simulation semantics in the spectrum we need $\mathcal{L}'_{TS}$, that will be based on $\mathcal{L}'_T$, to be defined in the next section.

The definition above differs from the particular case of ready simulation in Definition 6.3 in the two first rules, by means of which we impose that the process will traverse states which are in the corresponding N-equivalence class all along the tree of computations checked by a formula in $\mathcal{L}'_{NS}$. Note that the combination of positive and negated formulas allows us to shape each of these classes. Next we state the equivalence between our logics for the simulation semantics and those by van Glabbeek's recalled in Table 3.

Proposition 6.11. *(1) $\mathcal{L}'_S \sim \mathcal{L}_S$; (2) $\mathcal{L}'_{CS} \sim \mathcal{L}_{CS}$; and (3) $\mathcal{L}'_{2S} \sim \mathcal{L}_{2S}$.*

Proof.

(1) The clauses defining $\mathcal{L}'_S$ and $\mathcal{L}_S$ produce the same set of formulas. The first two clauses in $\mathcal{L}'_S$ only add the two trivial formulas $\top$ and $\neg\top$ because in $\mathcal{L}'_U = \{\top\}$.
(2) Again, the sets of formulas produced by $\mathcal{L}'_{CS}$ and $\mathcal{L}_{CS}$ are the same because the two first clauses of L'_{CS} can only generate $\top$, $\neg\top$, 0 and $\neg 0$ from $\mathcal{L}'_C = \{\top, \neg\top\}$. 0 is needed to reflect the second clause in the definition of $\mathcal{L}_{CS}$, while $\neg 0 \equiv \bigvee_{a\in Act} a\top$ so that any formula containing $\neg 0$ can be rewritten into a disjunction of formulas in $\mathcal{L}_{CS}$.

(3) Once again, the sets generated by $\mathcal{L}'_{2S}$ and $\mathcal{L}_{2S}$ are the same. The clause "if $\sigma \in \mathcal{L}'_S$ then $\sigma \in \mathcal{L}'_{2S}$" in $\mathcal{L}'_{2S}$ does not generate any new formulas because $\mathcal{L}_S \subseteq \mathcal{L}_{2S}$ (the formulas in $\mathcal{L}_S$ are exactly those that can be created using only the last two clauses in the definition of $\mathcal{L}_{2S}$).

□

Remark 6.12. We can use both positive formulas in $\mathcal{L}'_C$ and their negations for defining $\mathcal{L}'_{CS}$ due to the fact that C-constrained simulation can be built from the equivalence relation defined by C as constraint. However, we could also use $\sqsubseteq_C$ as a constraint and then remove the clause "if $\sigma \in \mathcal{L}'_C$ then $\sigma \in \mathcal{L}'_{S_C}$", which generates $\neg 0 \in \mathcal{L}_{S_C}$. The other clause, which generates $0 \in \mathcal{L}'_{S_C}$, is crucial and cannot be removed from the definition. These two facts also concur in the definition of the other simulation semantics in the extended spectrum, for which we also present a logical characterization including the two clauses above.

6.2.2. *Logical characterization of the linear semantics.* We start by defining the closure operators by means of which we express the extent to which conjunction and negation can be used in the logical characterizations of each of the linear semantics.

Definition 6.13. Given a logical set $\mathcal{L}'_N$ with $N \in \{U, C, I, T, S\}$, we define:

(1) Its *symmetric closure* $\mathcal{L}'^{\overline{=}}_N$ by: if $\sigma \in \mathcal{L}'_N$ then $\sigma \in \mathcal{L}'^{\overline{=}}_N$ and $\neg\sigma \in \mathcal{L}'^{\overline{=}}_N$; if $\sigma_i \in \mathcal{L}'^{\overline{=}}_N$ for all $i \in I$ then $\bigwedge_{i\in I} \sigma_i \in \mathcal{L}'^{\overline{=}}_N$.
(2) Its *negative closure* $\mathcal{L}'^{\neg}_N$ by: if $\sigma \in \mathcal{L}'_N$ then $\neg\sigma \in \mathcal{L}'^{\neg}_N$; if $\sigma_i \in \mathcal{L}'^{\neg}_N$ for all $i \in I$ then $\bigwedge_{i\in I} \sigma_i \in \mathcal{L}'^{\neg}_N$.
(3) Its *positive closure* $\mathcal{L}'^{\checkmark}_N$ by: if $\sigma \in \mathcal{L}'_N$ then $\sigma \in \mathcal{L}'^{\checkmark}_N$; if $\sigma_i \in \mathcal{L}'^{\checkmark}_N$ for all $i \in I$ then $\bigwedge_{i\in I} \sigma_i \in \mathcal{L}'^{\checkmark}_N$.

Remark 6.14. Obviously these closures make sense for any given logic $\mathcal{L}$, but we prefer to restrict our attention to $\mathcal{L}'_N$ since it will be enough for our goal and gives rise to a simpler notation.

Whenever we have a bag of "good" properties (such as $\mathcal{L}'_N$ above), to assert by means of a single formula which is the subset of properties that a certain element satisfies it is not enough to assert that it satisfies each of them: we also need to assert that it does not satisfy any of the rest. This is why we need formulas in the symmetric closure. By contrast, if the only available formulas belong to the negative (resp. positive) closure, we can only assert that the element has at most (resp. at least) the enumerated properties. Next we present the unified logics for all the linear semantics in the spectrum.

Definition 6.15. Inspired by the orders $\leq^{l}_N$, $\leq^{l\supseteq}_N$, $\leq^{lf}_N$, and $\leq^{lf\supseteq}_N$, we define the set of formulas $\mathcal{L}'_{\leq^{l}_N}$, $\mathcal{L}'_{\leq^{l\supseteq}_N}$, $\mathcal{L}'_{\leq^{lf}_N}$, and $\mathcal{L}'_{\leq^{lf\supseteq}_N}$, respectively, by means of the rules:

(1)
- $\top \in \mathcal{L}'_{\leq^{l}_N}$;
- if $\varphi \in \mathcal{L}'_{\leq^{l}_N}$ and $\sigma \in \mathcal{L}'^{\overline{=}}_N$ then $\sigma \wedge \varphi \in \mathcal{L}'_{\leq^{l}_N}$;
- if $\varphi \in \mathcal{L}'_{\leq^{l}_N}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{\leq^{l}_N}$.

(2)
- $\top \in \mathcal{L}'_{\leq^{l\supseteq}_N}$;
- if $\varphi \in \mathcal{L}'_{\leq^{l\supseteq}_N}$ and $\sigma \in \mathcal{L}'^{\neg}_N$ then $\sigma \wedge \varphi \in \mathcal{L}'_{\leq^{l\supseteq}_N}$;

- if $\varphi \in \mathcal{L}'_{\leq_N^{l\supseteq}}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{\leq_N^{l\supseteq}}$.

(3)
- $\top \in \mathcal{L}'_{\leq_N^{lf}}$;
- if $\sigma \in \mathcal{L}_N^{\equiv}$ then $\sigma \in \mathcal{L}'_{\leq_N^{lf}}$;
- if $\varphi \in \mathcal{L}'_{\leq_N^{lf}}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{\leq_N^{lf}}$.

(4)
- $\top \in \mathcal{L}'_{\leq_N^{lf\supseteq}}$;
- if $\sigma \in \mathcal{L}_N^{\neg}$ then $\sigma \in \mathcal{L}'_{\leq_N^{lf\supseteq}}$;
- if $\varphi \in \mathcal{L}'_{\leq_N^{lf\supseteq}}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{\leq_N^{lf\supseteq}}$.

Note that for the coarsest semantics (i.e. those corresponding to plain refusals and plain readiness when $N = I$) we only check for N at the "end" of the formula because there are no conjunctions in the corresponding languages $\mathcal{L}'_{\leq_N^{lf}}$ and $\mathcal{L}'_{\leq_N^{lf\supseteq}}$, except for those stemming from the corresponding closures $\mathcal{L}_N^{\equiv}$ and $\mathcal{L}_N^{\neg}$. The other two logics do introduce additional conjunctions that allow to observe N along the computations.

We have used the negative and symmetric closures for the "failures-based" and"readies-based" semantics, and we can use the positive closure to define two new semantics that have not been considered earlier in this paper, nor elsewhere as far as we know. For that we need to observe partial offers along a computation, or just at its end, where X is a partial offer of p if $X \subseteq I(p)$. It is clear the duality with respect to the failures semantics, where F is a failure of p if $I(p) \subseteq \overline{F}$. We can introduce these two new semantics at each layer of the spectrum through the corresponding partial offers for each $N \in \{U, C, I, T, S\}$.

Definition 6.16.

(1) The semantics of *partial offer traces* for the constraint N is that defined by the logic $\mathcal{L}'_{\leq_N^{l\subseteq}}$ with:
- $\top \in \mathcal{L}'_{\leq_N^{l\subseteq}}$;
- if $\varphi \in \mathcal{L}'_{\leq_N^{l\subseteq}}$ and $\sigma \in \mathcal{L}_N^{\surd}$ then $\sigma \wedge \varphi \in \mathcal{L}'_{\leq_N^{l\subseteq}}$;
- if $\varphi \in \mathcal{L}'_{\leq_N^{l\subseteq}}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{\leq_N^{l\subseteq}}$.

(2) The semantics of *partial offers* for the constraint N is that defined by the logic $\mathcal{L}'_{\leq_N^{lf\subseteq}}$ with:
- $\top \in \mathcal{L}'_{\leq_N^{lf\subseteq}}$;
- if $\sigma \in \mathcal{L}_N^{\surd}$ then $\sigma \in \mathcal{L}'_{\leq_N^{lf\subseteq}}$;
- if $\varphi \in \mathcal{L}'_{\leq_N^{lf\subseteq}}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{\leq_N^{lf\subseteq}}$.

Duality between failures and partial offers causes the picture of the complete layer of linear semantics for each N to become two diamonds that share the side corresponding to the readies-based semantics. Now, recalling Theorem 6.7.

Proposition 6.17.

(1) *$\mathcal{L}'_F$ and $\mathcal{L}'_{\leq_I^{lf\subseteq}}$ are incomparable: $p \leq_I^{lf\supseteq} q$ does not imply $p \leq_I^{lf\subseteq} q$ and $p \leq_I^{lf\subseteq} q$ does not imply $p \leq_I^{lf\supseteq} q$.*

(2) $\mathcal{L}'_{FT}$ *and* $\mathcal{L}'_{\leq_I^{l\subseteq}}$ *are incomparable:* $p \leq_I^{l\supseteq} q$ *does not imply* $p \leq_I^{l\subseteq} q$ *and* $p \leq_I^{l\subseteq} q$ *does not imply* $p \leq_I^{l\supseteq} q$.

Proof. In fact, we have a stronger result by combining these two statements: if we consider $p = ab + ac$, $q = a(b + c)$, and $r = p + q$, then $p =_I^{l\supseteq} r$ but $r \not\leq_I^{lf\subseteq} p$, and $q =_I^{l\subseteq} r$ but $r \not\leq_I^{lf\supseteq} q$. □

Similar counterexamples exist for $N \in \{T, S\}$. However, for $N \in \{U, C\}$, which produce the trace and the completed trace semantics, respectively, it is easy to prove that the six logics of the layer are equivalent.

Proposition 6.18.
(1) $\mathcal{L}'_{\leq_U^{lf}} = \mathcal{L}'_{\leq_U^{l}} = \mathcal{L}'_{\leq_U^{l\supseteq}} = \mathcal{L}'_{\leq_U^{l\subseteq}} = \mathcal{L}'_{\leq_U^{lf\supseteq}} = \mathcal{L}'_{\leq_U^{lf\subseteq}} = \mathcal{L}_T$
(2) $\mathcal{L}'_{\leq_C^{lf\supseteq}} = \mathcal{L}'_{\leq_C^{lf\subseteq}} = \mathcal{L}'_{\leq_C^{l\supseteq}} = \mathcal{L}'_{\leq_C^{l\subseteq}} = \mathcal{L}'_{\leq_C^{lf}} = \mathcal{L}'_{\leq_C^{l}} = \mathcal{L}_{CT}$.

Proof.
(1) Trivial, since the sets of clauses defining $\mathcal{L}'_{\leq_U^{lf}}$ and $\mathcal{L}_T$ are almost the same. Note that the clause "if $\sigma \in \mathcal{L}_U^{\equiv}$ then $\sigma \in \mathcal{L}'_{\leq_U^{lf}}$" does not give rise to new formulas because $\mathcal{L}_U^{\equiv} = \{\top\}$.
(2) Note that the sets of clauses defining $\mathcal{L}'_{\leq_C^{lf\supseteq}}$ and $\mathcal{L}_{CT}$ are the same but for the clause "if $\sigma \in \mathcal{L}_C^{\neg}$ then $\sigma \in \mathcal{L}'_{\leq_C^{lf\supseteq}}$". On the one hand, this causes $\neg\top \in \mathcal{L}'_{\leq_C^{lf\supseteq}}$ (which adds nothing) because $\top \in \mathcal{L}'_C$ and thus $\neg\top \in \mathcal{L}_C^{\neg}$. On the other hand, we also have $0 \in \mathcal{L}'_{\leq_C^{lf\supseteq}}$ because $\neg 0 \in \mathcal{L}'_C$ and then $\neg\neg 0 \in \mathcal{L}_C^{\neg}$. □

Corollary 6.19. $\mathcal{L}'_{\leq_U^{lf}} \sim \mathcal{L}_T$ *and* $\mathcal{L}'_{\leq_C^{lf\supseteq}} \sim \mathcal{L}_{CT}$.

An interesting result illustrating the generality of our characterizations concerns one of the finest semantics in the classic spectrum: possible futures. Possible futures is located in Figure 1 below 2-nested simulation because the more accurate trace simulation semantics was not yet included in the spectrum; this is corrected in the spectrum in Figure 11. Indeed, for $N = T$ we have the following result.

Proposition 6.20. $\mathcal{L}'_{\leq_T^{lf}} = \mathcal{L}_{PF}$.

Proof. Trivial, since the sets of clauses defining $\mathcal{L}'_{\leq_T^{lf}}$ and $\mathcal{L}_{PF}$ are almost the same: our definition includes the clause "$\top \in \mathcal{L}'_{\leq_T^{lf}}$", which does not appear explicitly in that of $\mathcal{L}_{PF}$ because it corresponds to the conjunction of an empty set of formulas. □

Corollary 6.21. $\mathcal{L}'_{\leq_T^{lf}} \sim \mathcal{L}_{PF}$.

6.2.3. *Logical characterization of the deterministic branching semantics.* Now we consider the deterministic branching semantics. In the classic spectrum the only such semantics is possible worlds but, as we pointed out before, there is one such semantics at each layer of the extended spectrum.

Formulas \ Constraints ($\mathcal{N}$)	U	C	I	T	S	B
$\top \in \mathcal{L}'_{\mathcal{N}}$	•	•	•	•	ν	ν
$\neg\top = \bot \in \mathcal{L}'_{\mathcal{N}}$	ν	ν	ν	ν	ν	ν
$\neg 0 \in \mathcal{L}'_{\mathcal{N}}$		•	•	ν	ν	ν
$a \in Act \Rightarrow\ a\top \in \mathcal{L}'_{\mathcal{N}}$			•	ν	ν	ν
$\varphi \in \mathcal{L}'_{\mathcal{N}},\ a \in Act \Rightarrow$ $a\varphi \in \mathcal{L}'_{\mathcal{N}}$				•	•	•
$\varphi_i \in \mathcal{L}'_{\mathcal{N}}\ \forall i \in I \Rightarrow$ $\bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{\mathcal{N}}$					•	•
$\varphi \in \mathcal{L}'_{\mathcal{N}} \Rightarrow$ $\neg\varphi \in \mathcal{L}'_{\mathcal{N}}$						•

Table 4: Logical characterizations of the semantics used as constraints.

In order to capture determinism we need to consider conjunctive formulas to express the desired branching, but only when it corresponds to a choice between different actions. This leads us to the following scheme:

$$\text{if } X \subseteq Act \text{ and } \varphi_a \in \mathcal{L}_{D_N} \text{ for all } a \in X, \text{ then } \bigwedge_{a\in X} a\varphi_a \in \mathcal{L}_{D_N}.$$

Definition 6.22. For each $N \in \{U, C, I, T, S\}$, we define the formulas of $\mathcal{L}'_{D_N}$ by:

- $\top \in \mathcal{L}'_{D_N}$;
- if $\varphi \in \mathcal{L}'_{D_N}$ and $\sigma \in \mathcal{L}^{\equiv}_N$ then $\sigma \wedge \varphi \in \mathcal{L}'_{D_N}$;
- if $X \subseteq Act$ and $\varphi_a \in \mathcal{L}'_{D_N}$ for all $a \in X$ then $\bigwedge_{a\in X} a\varphi_a \in \mathcal{L}'_{D_N}$.

For $N = I$ we obtain the unified logical characterization of the possible worlds semantics.

Proposition 6.23. $\mathcal{L}'_{D_I} \supseteq \mathcal{L}_{PW}$.

Proof. Analogous to the case of ready simulation semantics. □

Proposition 6.24. $\mathcal{L}'_{D_I} \nsim \mathcal{L}_{PW}$.

Proof. This is a consequence of the fact that the original logical characterization of the possible worlds semantics, $\mathcal{L}_{PW}$, was wrong. For instance, taking $p = abc + a(bc + d) + ab$ and $q = a(bc + d) + ab$ then $p \not\equiv_{PW} q$ but $p \sim_{\mathcal{L}_{PW}} q$, since $\mathcal{L}_{PW}$ cannot "observe" the intermediate offer that makes the possible world abc different from those of q. By contrast, the formula $\varphi = a(\neg d \wedge bc) \in \mathcal{L}'_{D_I}$ is enough to distinguish p and q, since $p \models \varphi$ and $q \not\models \varphi$. □

We postpone to Section 7 the proof of the equivalence between our observational and logical characterizations of the possible worlds semantics. As a consequence of this correspondence, we have that a logical characterization only works in the infinite case if we restrict ourselves to image-finite processes.

In Tables 4 and 5 we present our results in a three-dimensional way. Table 5 shows the rules defining the logics characterizing each of the semantics at each layer of the spectrum. On top of it also appears, as example, the classic notation for the corresponding semantics represented when $N = I$. Table 4 contains the logics that characterize the constraint governing each of these layers. There are two semantics that are included in both tables, in order to emphasize their double role as "main" and "auxiliary" semantics. However they are disguised under different names: this is the case of $T = \leq^l_U$ (in fact, it is also equal to the other three linear U-semantics) and $S = US$.

Formulas \ Semantics ($\mathcal{Y}_\mathcal{N}$)	$\leq_N^{\mathbf{lf}\supseteq}$	$\leq_N^{\mathbf{lf}}$	$\leq_N^{\mathbf{l}\supseteq}$	$\leq_N^{\mathbf{l}}$	D_N	NS	$N \in \{U,C,I,T,S\}$
	F	R	FT	RT	PW	RS	when $N = I$
$\top \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$	•	•	•	•	•	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$, $a \in Act \Rightarrow$ $a\varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$	•	•	•	•	ν	•	
$\varphi \in \mathcal{L}_N^{\supseteq} \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$	•	ν	ν	ν	ν	ν	
$\varphi \in \mathcal{L}_N^{=} \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$		•		ν	ν	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$, $\sigma \in \mathcal{L}_N^{\supseteq} \Rightarrow$ $\sigma \wedge \varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$			•	ν	ν	ν	
$\varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$, $\sigma \in \mathcal{L}_N^{=} \Rightarrow$ $\sigma \wedge \varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$				•	•	ν	
$X \subseteq Act$, $\varphi_a \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$ $\forall a \in X \Rightarrow$ $\bigwedge_{a\in X} a\varphi_a \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$					•	ν	
$\varphi_i \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$ $\forall i \in I \Rightarrow$ $\bigwedge_{i\in I} \varphi_i \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$						•	
$\varphi \in \mathcal{L}_N \Rightarrow$ $\varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$						•	
$\varphi \in \mathcal{L}_N \Rightarrow$ $\neg\varphi \in \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$						•	

Table 5: Our new logical characterizations for the semantics at each level of the spectrum.

7. Relating the unified logics and the unified observational model

In this section we will relate our unified logical characterizations and the unified observational semantics. As indicated in Section 2, we have to restrict ourselves to image-finite processes; as a byproduct, the finite parts of each of the corresponding languages, that are obtained by intersection with $\mathcal{L}^f_{HM}$, provide us with a pure finite logical characterization of the semantics. However, it is convenient in the first part of this Section to consider still the full (infinitary) logic characterizing each of the semantics.

Definition 7.1 (**Normal formulas $\mathcal{N}(\mathcal{L})$**)**.**

(1) Given a set of formulas $\mathcal{L}$ whose outermost operator is not conjunction, the set $\mathcal{N}(\mathcal{L})$ of induced *normal formulas* is defined by:
- $\top \in \mathcal{N}(\mathcal{L})$;
- if $\Gamma_1, \Gamma_2 \subseteq \mathcal{L}, \{a_i \mid i \in I\} \subseteq Act$, and $\varphi_i \in \mathcal{N}(\mathcal{L})$, then $(\bigwedge_{\sigma\in\Gamma_1} \sigma \wedge \bigwedge_{\sigma\in\Gamma_2} \neg\sigma) \wedge \bigwedge_{i\in I} a_i\varphi_i \in \mathcal{N}(\mathcal{L})$.

(2) For each $N \in \{U,C,I,T,S\}$ and each $\mathcal{Y}_\mathcal{N} \in \{NS, \leq_N^{l}, \leq_N^{l\supseteq}, \leq_N^{lf}, \leq_N^{lf\supseteq}, \leq_N^{l\subseteq}, \leq_N^{lf\subseteq}, D_N\}$ in the spectrum, we define the set of normal formulas $\mathcal{N}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}''_\mathcal{N}) \subseteq \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$ as $\mathcal{N}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}''_\mathcal{N}) = \mathcal{N}(\mathcal{L}''_\mathcal{N}) \bigcap \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$, where $\mathcal{L}''_N$ is the set of formulas in $\mathcal{L}'_N$ whose outermost operator is not conjunction.

Remark 7.2. The clause in Definition 7.1.1 is more involved than it appears. Initially, we can apply it with $I = \emptyset$ to obtain the first (non-trivial) normal formulas and then recursively to obtain more complex normal formulas; note that the two first subformulas stem always from the original set $\mathcal{L}$. By abuse of notation, when some of the elements in our normal formulas do not appear in the corresponding set $\mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$, we assume that these formulas have been extended by conjunction with $\top$ using the fact that $\bigwedge_{\sigma\in\emptyset} \sigma$ is another syntactic form to express $\top$.

Also note that infinite conjunction is allowed in the two first subformulas. As a consequence, if we consider the tree-like form of these (possibly infinitary) formulas they could have infinite depth. However, if we define the normal depth of formulas in $\mathcal{N}(\mathcal{L}_{\mathcal{N}})$ as that obtained by counting the recursive nesting in the application of Definition 7.1, then any normal formula has finite normal depth, and the set they form can be explored by structural induction.

Theorem 7.3. *Each set of normal formulas $\mathcal{N}_{\mathcal{Y}_{\mathcal{N}}}(\mathcal{L}''_{\mathcal{N}})$ associated to the semantics in the spectrum is equivalent to the full set of formulas $\mathcal{L}'_{\mathcal{Y}_{\mathcal{N}}}$.*

Proof. By structural induction, all the formulas in $\mathcal{L}'_{\mathcal{Y}_{\mathcal{N}}}$ admit a normal formula in the sense of Definition 7.1, that is obtained by gathering the subformulas and applying Proposition 6.2. □

Definition 7.4. The set of *complete normal* formulas $\mathcal{CN}(\mathcal{L})$ (resp., the set of complete normal formulas associated to each semantics in the spectrum, $\mathcal{CN}_{\mathcal{Y}_{\mathcal{N}}}(\mathcal{L}''_{\mathcal{N}})$) is the set of normal formulas (resp., the set of normal formulas associated to each semantics in the spectrum) for which the rule in Definition 7.1 is applied with $\Gamma_2 = \overline{\Gamma_1}$.

Now we prove that infinite conjunction in Definition 7.1 can be approximated by finite conjunction.

Theorem 7.5. *If we restrict ourselves to image-finite processes, for each denumerable set of formulas $\mathcal{L}$, any complete normal formula $\varphi \in \mathcal{CN}(\mathcal{L})$ can be approximated by a set of finite normal formulas $\{\varphi^k \mid k \in \mathbb{N}\}$ that only use finite conjunction, that is, $p \models \varphi$ iff $p \models \varphi^k$ for all $k \in \mathbb{N}$.*

Proof. We define the sequence φ^k by structural induction on the normal depth of φ:

- $\varphi = (\bigwedge_{\sigma\in\Gamma_1} \sigma \wedge \bigwedge_{\sigma\in\overline{\Gamma_1}} \neg\sigma)$. We consider a fixed enumeration of the set $\mathcal{L} = \{\sigma_n \mid n \in \mathbb{N}\}$, and define $\mathcal{L}^{\leqslant n} = \{\sigma_j \in \mathcal{L} \mid j \leqslant n\}$. Then, for each $k \in \mathbb{N}$:
$$\varphi^k = \bigwedge_{\sigma\in\Gamma_1\cap\mathcal{L}^{\leqslant k}} \sigma \wedge \bigwedge_{\sigma\in\overline{\Gamma_1}\cap\mathcal{L}^{\leqslant k}} \neg\sigma.$$
We have $p \models \varphi \Leftrightarrow (p \models \sigma\ \forall\sigma \in \Gamma_1$ and $p \not\models \sigma\ \forall\sigma \notin \Gamma_1)$ and $p \models \varphi^k\ \forall k \in \mathbb{N} \Leftrightarrow (p \models \sigma\ \forall\sigma \in \Gamma_1 \cap \mathcal{L}^{\leqslant k}$ and $p \not\models \sigma\ \forall\sigma \in \overline{\Gamma_1} \cap \mathcal{L}^{\leqslant k})$ and the result follows from a the equality
$$\Gamma_1 = \Gamma_1 \cap \mathcal{L} = \Gamma_1 \cap (\bigcup_{k\in\mathbb{N}} \mathcal{L}^{\leqslant n}).$$
- $\varphi = (\bigwedge_{\sigma\in\Gamma_1} \sigma \wedge \bigwedge_{\sigma\in\overline{\Gamma_1}} \neg\sigma) \wedge \bigwedge_{i\in I} a_i\varphi_i$. By structural induction we can assume that the result is true for any subformula φ_i. Then we define $\varphi^k = \bigwedge_{\sigma\in\Gamma_1\cap\mathcal{L}^{\leqslant k}} \sigma \wedge \bigwedge_{\sigma\in\overline{\Gamma_1}\cap\mathcal{L}^{\leqslant k}} \neg\sigma \wedge \bigwedge_{i\in I} a_i\varphi_i^k$. Now, if we decompose φ as $\varphi_I \wedge \varphi_{II}$ (taking $\varphi_{II} = \bigwedge_{i\in I} a_i\varphi_i$, and analogously for the set of approximations) we have that $p \models \varphi^k$ iff $p \models \varphi_I^k$ and $p \models \varphi_{II}^k$. If $p \models \varphi^k$ then $p \models \varphi_I^k$ for all $k \in \mathbb{N}$ and arguing as in the base case above we conclude that $p \models \varphi_I$. Any image-finite process p can be decomposed as $p = \sum_{a_i\in Act} \sum_{j=1}^{m_i} a_i^j p_i^j$, and we have $p \models \varphi_{II}^k$ iff for all i there exists j with $a_i = a_i^j$ and $p_i^j \models \varphi_i^k$. Then, if $p \models \varphi_{II}^k$ for all $k \in \mathbb{N}$, for each i there exists some $j \in 1..m_i$ such that $p_i^j \models \varphi_i^k$ for infinitely many k, but this means that $p_i^j \models \varphi_i^k$ for all $k \in \mathbb{N}$ and then, by the induction hypothesis, $p_i^j \models \varphi_i$ thus getting $p \models \varphi$.

□

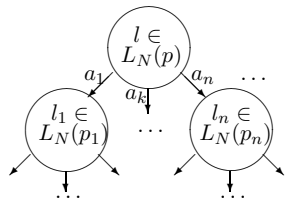


Figure 14: A branching observation.

Definition 7.6. For each $N \in \{U, C, I, T, S\}$ and each $\mathcal{Y}_\mathcal{N} \in \{NS, \leq_N^l, \leq_N^{l\supseteq}, \leq_N^{lf}, \leq_N^{lf\supseteq}, \leq_N^{l\subseteq}, \leq_N^{lf\subseteq}, D_N\}$ in the spectrum, we define the finite logic for the semantics $\mathcal{L}^f_{\mathcal{Y}_\mathcal{N}}$ as $\mathcal{L}'_{\mathcal{Y}_\mathcal{N}} \cap \mathcal{L}^f_{HM}$.

Corollary 7.7. *For each $N \in \{U, C, I, T, S\}$ and each $\mathcal{Y}_\mathcal{N} \in \{NS, \leq_N^l, \leq_N^{l\supseteq}, \leq_N^{lf}, \leq_N^{lf\supseteq}, \leq_N^{l\subseteq}, \leq_N^{lf\subseteq}, D_N\}$ in the spectrum, if we restrict ourselves to the set of image-finite processes we have $\mathcal{L}^f_{\mathcal{Y}_\mathcal{N}} \sim \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$.*

Proof. We only need to apply Theorem 7.5. The only non trivial case is when $N = S$, where we have to apply twice the Theorem, using also the fact that $CN(\mathcal{L}''_S) \sim CN(\mathcal{L}^f_S)$, because $\mathcal{L}''_S \sim \mathcal{L}^f_S$. □

Theorem 7.8. *For each $N \in \{U, C, I, T, S\}$ and each $\mathcal{Y}_\mathcal{N} \in \{NS, \leq_N^l, \leq_N^{l\supseteq}, \leq_N^{lf}, \leq_N^{lf\supseteq}, \leq_N^{l\subseteq}, \leq_N^{lf\subseteq}, D_N\}$ in the spectrum there exists a correspondence between the set of complete normal formulas $\mathcal{CN}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}''_\mathcal{N})$ and the corresponding domain of observations ΩGO_N with $\Omega \in \{B, L\}$. This correspondence $\leftrightarrow$ satisfies that $\varphi \leftrightarrow \theta$ implies that $(p \models \varphi$ iff $\theta \in \Omega GO_N(p))$. Moreover:*

(1) *The set of complete normal formulas $\mathcal{CN}_{\mathcal{NS}}(\mathcal{L}''_\mathcal{N})$ (resp. $\mathcal{CN}_{\mathcal{D}_\mathcal{N}}(\mathcal{L}''_\mathcal{N})$) and the domain of branching general observations BGO_N (resp. $dBGO_N$) are isomorphic, that is, $\leftrightarrow$ is one to one.*
(2) *The set of complete normal formulas $\mathcal{CN}_{\leq_N^l}(\mathcal{L}''_\mathcal{N})$, $\mathcal{CN}_{\leq_N^{l\supseteq}}(\mathcal{L}''_\mathcal{N})$ and the domain of linear general observations LGO_N are isomorphic, that is, $\leftrightarrow$ is one to one.*
(3) *The set of complete normal formulas $\mathcal{CN}_{\leq_N^{lf}}(\mathcal{L}''_\mathcal{N})$ (resp. $\mathcal{CN}_{\leq_N^{lf\supseteq}}(\mathcal{L}''_\mathcal{N})$) and the quotient domain $LGO_N/_{\simeq_N^{lf}}$ (resp. $LGO_N/_{\simeq_N^{lf\supseteq}}$) are isomorphic, that is, $\leftrightarrow^{-1}$ is injective and $\varphi \leftrightarrow \theta$ iff $\theta \simeq_N^{lf\supseteq} \theta_\varphi$, for some adequate θ_φ.*

Proof.

(1) As can be seen in Figure 14, a branching observation is a labeled tree whose nodes are local observations and whose arcs are labeled by actions.

The general form of any complete normal formula in $\mathcal{CN}_{\mathcal{NS}}(\mathcal{L}_\mathcal{N})$ is $(\bigwedge_{\sigma\in\Gamma} \sigma \wedge \bigwedge_{\sigma\notin\Gamma} \neg\sigma) \wedge \bigwedge_{i\in I} a_i\varphi_i$, with $\varphi_i \in \mathcal{CN}_{\mathcal{NS}}(\mathcal{L}_\mathcal{N})$ for all $i \in I$. Since the language $\mathcal{L}'_N$ characterizes the semantics used to get the local observations, we can associate to each complete formula $(\bigwedge_{\sigma\in\Gamma} \sigma \wedge \bigwedge_{\sigma\notin\Gamma} \neg\sigma)$ the corresponding local observation $l \in L_N$. Then, by structural induction, we obtain the observation associated to each formula $\varphi_i \in \mathcal{CN}_{\mathcal{NS}}(\mathcal{L}_\mathcal{N})$, thus getting the branching general observation BGO_N associated to the given formula. It is easy to see that this correspondence is indeed a bijection.

The case for $\mathcal{CN}_{\mathcal{D}_\mathcal{N}}(\mathcal{L}_\mathcal{N})$ is analogous, but now it is not allowed to have repeated actions in the arcs leaving any node of an observation; this is obviously reflected in the form of the formulas in the corresponding language.

(2) The case for $\mathcal{CN}_{\leq_N^l}(\mathcal{L}_\mathcal{N})$ is similar to the previous one, but now the obtained (degenerated) tree is just a single branch corresponding to a lgo in LGO_N.

For $\mathcal{CN}_{\leq_N^{l\supseteq}}(\mathcal{L}_\mathcal{N})$ the general form of a complete normal formula is $\varphi = (\top \wedge \bigwedge_{\sigma\notin\Gamma} \neg\sigma) \wedge a\varphi'$, with $\varphi' \in \mathcal{CN}_{\leq_N^{l\supseteq}}(\mathcal{L}_\mathcal{N})$. If we close the set Γ by derivability obtaining Γ' and then consider its complement $\overline{\Gamma'}$, we can consider the local observation l that satisfies all the formulas in $\overline{\Gamma'}$ and none in Γ'. The linear general observation lgo corresponding to φ is then recursively defined as $\langle l, \{(a, lgo')\}\rangle$ where lgo' is the linear general observation corresponding to φ'.

To proceed in the opposite direction, we just need to take as Γ the complement of the set of formulas in $\mathcal{L}'_N$ satisfied by the local observation l at the root of the given LGO_N, and then proceed in a recursive way.

(3) In this case, the general form of a complete normal formula in $\mathcal{CN}_{\leq_N^{lf}}(\mathcal{L}_\mathcal{N})$ is $\varphi = \top \wedge a_1(\dots(\top \wedge a_{n-1}(\top \wedge a_n(\bigwedge_{\sigma\in\Gamma} \sigma \wedge \bigwedge_{\sigma\notin\Gamma} \neg\sigma)\dots)$. Now we establish a correspondence between the set of local observations L_N and the sets $\Gamma \subseteq \mathcal{L}_N$ as done in cases (1) and (2) above, and then define the correspondence $\leftrightarrow$ by ignoring the values of all the intermediate local observations in the considered lgo, keeping only the local observation at the end.

For $\mathcal{CN}_{\leq_N^{lf\supseteq}}(\mathcal{L}_\mathcal{N})$ we just need to apply the same procedure above combined with the ideas along the proof for $\mathcal{CN}_{\leq_N^{l\supseteq}}(\mathcal{L}_\mathcal{N})$. □

Remark 7.9. It came as a surprise to notice that the $lgo's$ in LGO_N are in a bijective relation both with the complete normal formulas in $\mathcal{N}_{\leq_N^l}(\mathcal{L}_\mathcal{N})$ and those in $\mathcal{N}_{\leq_N^{l\supseteq}}(\mathcal{L}_\mathcal{N})$, so let us consider the case $N = I$ to explain this fact. A *cnf* in $\mathcal{N}_{\leq_I^l}(\mathcal{L}_\mathcal{I})$ specifies the corresponding local observation $I(p) \subseteq \mathcal{P}(Act)$ by means of a formula $(\bigwedge_{\sigma\in\Gamma} \sigma \wedge \bigwedge_{\sigma\notin\Gamma} \neg\sigma)$, where the formulas in Γ are just the elements of the corresponding set $I(p)$ while those in $\overline{\Gamma}$ correspond to its complement. When considering the failure trace semantics, the formulas in $\mathcal{N}_{\leq_I^{l\supseteq}}(\mathcal{L}_\mathcal{I})$ only contain the part $\bigwedge_{\sigma\notin\Gamma} \neg\sigma$ corresponding to the complement $\overline{I(p)}$. Since in this case the sets of lgo's could be assumed to be closed with respect to the order $\leq_N^{l\supseteq}$ in Definition 4.19, soundness is retained after "assuming" that any formula $\bigwedge_{\sigma\notin\Gamma} \neg\sigma$ "generates" the observation associated to Γ, even though some of the formulas $\sigma \in \Gamma$ may not be satisfied when the corresponding $I(p)$ is smaller. But for the failures and failure trace semantics we can proceed by closing the set of offers upwards with respect to $\subseteq$ and no new failure is introduced.

Theorem 7.10. *For each $N \in \{U, C, I, T, S\}$ and each $\mathcal{Y}_\mathcal{N} \in \{NS, \leq_N^l, \leq_N^{l\supseteq}, \leq_N^{lf}, \leq_N^{lf\supseteq}, \leq_N^{l\subseteq}, \leq_N^{lf\subseteq}, D_N\}$ in the spectrum, if we restrict ourselves to image-finite processes, the logical semantics $\sqsubseteq^f_{\mathcal{Y}_\mathcal{N}}$ induced by the logic $\mathcal{L}^f_{\mathcal{Y}_\mathcal{N}}$, is equivalent to the corresponding observational semantics in Definitions 4.2, 4.11 and 4.32. In order to unify our notation, here we will denote by GO_N the corresponding semantic domain.*

Proof. By Theorem 7.3, $\mathcal{L}'_{\mathcal{Y}_\mathcal{N}} \sim \mathcal{N}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}_\mathcal{N})$, and from Theorem 7.8 we get the isomorphism between the set $\mathcal{CN}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}_\mathcal{N})$ and the corresponding set of general observations GO_N.

To finish the proof, we just need to show that $\mathcal{N}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}_\mathcal{N})$ and $\mathcal{CN}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}_\mathcal{N})$ are equivalent. Any consistent formula in $\mathcal{N}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}_\mathcal{N})$ ($\Gamma_1 \bigcap \Gamma_2 = \emptyset$) provides only some partial information about the states in a computation, so that the concrete values of these states are characterized by a set Γ with $\Gamma_1 \subseteq \Gamma \subseteq \overline{\Gamma_2}$. Therefore, we can replace Γ_1 and Γ_2 by Γ and $\overline{\Gamma}$, respectively, adding the disjunction over all the possible values of Γ, to characterize the set of processes specified by the formula. Now it is enough to float the disjunction up to obtain a disjunction of formulas in $\mathcal{CN}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}_\mathcal{N})$, and applying Proposition 6.2 we get the equivalence between the two sets of formulas. Finally, we only need to apply Corollary 7.7 to conclude. □

Corollary 7.11.

(1) *The unified logical semantics in Definition 6.10 is equivalent to the N-simulation semantics.*
(2) *The unified logical semantics in Definition 6.15.1 is equivalent to the N-ready trace semantics.*
(3) *The unified logical semantics in Definition 6.15.2 is equivalent to the N-failure trace semantics.*
(4) *The unified logical semantics in Definition 6.15.3 is equivalent to the N-readiness semantics.*
(5) *The unified logical semantics in Definition 6.15.4 is equivalent to the N-failure semantics.*
(6) *The unified logical semantics in Definition 6.22 is equivalent to the N-deterministic branched semantics.*

Moreover, if we restrict ourselves to image-finite processes we have also an equivalence with the corresponding finite logical semantics.

Proof. Since it was proved in Section 4 that any observational semantics characterizes the corresponding (classical) semantics in the (extended) ltbt spectrum, the desired equivalence between our (unified) logical characterizations and the classical semantics is an immediate corollary. □

8. On the real diamond structure

This section is a practical proof of the suitability of our unification work. Some recently proposed semantics that were not in the original ltbt spectrum are nicely included in our extended spectrum, which shows why and how the old spectrum has to be expanded. Our unified approach immediately absorbs these new semantics and the results about the different characterizations are easily extended to cover them. We warmly thank Roscoe for pointing out to us his work on the stable revivals semantics [46, 48], where an endeavor for an adequate presentation of the notion of responsiveness for a CSP-like language is made. (Responsiveness had been previously studied by Fournet et al. in [27] for CCS, under the name of stuck-freeness.)

When faced with the diamond shape of the collection of linear semantics that are associated to each simulation semantics in the extended spectrum, it would be natural to expect it to reflect the structure of a lattice. Then, failure semantics would be the greatest lower bound of the readiness and failure trace semantics, while ready trace semantics would be the corresponding lowest upper bound. However, both intuitions turn out to be wrong

and a new semantics finer than failures and another one coarser than ready trace can be found: together with readiness and failure trace, they do constitute a lattice.

Let us first consider the case of the lowest upper bound. We postulate that the axiomatization of the associated semantics is obtained by instantiating our general axiom with the conjunction of the two conditions M_R and M_{FT}:

$$M_{R\wedge FT}(x,y,w) \iff I(x) \supseteq I(y) \text{ and } I(w) \subseteq I(y).$$

We denote with $\sqsubseteq_{R\wedge FT}$ the order axiomatized by the corresponding axiom $(ND^{R\wedge FT})$.

Definition 8.1. The *readiness and failure trace semantics*, or *join semantics* $R \wedge FT$, is that defined by the order $\sqsubseteq_{R\wedge FT}$ generated by the set of axioms $\{B_1\text{–}B_4, (RS), (ND^{R\wedge FT})\}$.

Proposition 8.2. *The ready trace semantics is strictly finer than the readiness and failure trace semantics.*

Proof. $\sqsubseteq_{RT} \subseteq \sqsubseteq_{R\wedge FT}$ is an immediate consequence of Proposition 3.2 and the fact that condition M_{RT} implies both M_R and M_{FT}, and hence also $M_{R\wedge FT}$. To show that $\sqsubseteq_{RT} \not\subseteq \sqsubseteq_{R\wedge FT}$, let us take $w = \mathbf{0}$, $y = b$, and $x = bB' + c$; then we have:

$$\underbrace{a(bB + bB' + c)}_{p} \sqsubseteq_{R\wedge FT} \underbrace{a(bB' + c) + abB}_{q}$$

but, if $I(B) \neq I(B')$,

$$a(bB + bB' + c) \not\sqsubseteq_{RT} a(bB' + c) + abB$$

because $\{a\}a\{b,c\}bI(B) \in \mathit{ReadyTraces}(p) \setminus \mathit{ReadyTraces}(q)$. □

It is clear that the readiness and failure trace semantics is finer than both the readiness and the failure trace semantics; to show that it is actually the coarsest upper bound we need to prove that $\sqsubseteq_{R\wedge FT} = \sqsubseteq_R \cap \sqsubseteq_{FT}$. Even if the axiom $(ND^{R\wedge FT})$ was created with this goal in mind, this cannot be easily shown using only algebraic arguments. Instead, it is trivial to obtain the observational characterization of the desired semantics by gathering together the failure trace and the ready observations. Based on Definition 4.19, we can define the corresponding order $\leq_N^{l\supseteq\wedge f}$ by taking

$$\mathcal{T} \leq_N^{l\supseteq\wedge f} \mathcal{T}' \iff \mathcal{T} \leq_N^{l\supseteq} \mathcal{T}' \text{ and } \mathcal{T} \leq_N^{lf} \mathcal{T}'.$$

A direct characterization can be obtained as follows. We combine both kinds of observations into a single family of decorated traces that we call *failure trace with final ready sets*, by considering failure sets all along the trace except at the end of it, where we introduce the corresponding ready set.

Definition 8.3. We define the order $\leq_N^{l\supseteq\wedge f}$ by

$$\mathcal{T} \leq_N^{l\supseteq\wedge f} \mathcal{T}' \iff \begin{array}{l}\text{for all } X_0a_1\dots X_n \in \mathcal{T} \text{ there is some } Y_0a_1\dots Y_n \in \mathcal{T}' \\ \text{with } X_n = Y_n \text{ and } X_i \supseteq Y_i, \text{ for } i \in 0..n-1.\end{array}$$

Proposition 8.4. *The semantics defined by the order $\sqsubseteq_{R\wedge FT}$ coincides with that defined by $\leq_I^{l\supseteq\wedge f}$ and is thus the lowest upper bound of the readiness and failure trace semantics.*

Proof. Similar to that of Theorem 5.4. □

Let us finally consider the logical characterization of this semantics. It is clear that the conjunction of two semantics should be characterized in a logical way by simply considering the union of the logics that characterize both semantics (although there could possibly be a more compact presentation).

Definition 8.5. We define the set of formulas $\mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$ as that generated by the clauses:

- $\top \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$;
- if $\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$ and $\sigma \in \mathcal{L}_I^{\neg}$ then $\sigma \wedge \varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$;
- if $\sigma \in \mathcal{L}_I^{\equiv}$ then $\sigma \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$;
- if $\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$.

Proposition 8.6. *The logical semantics $\sqsubseteq'_{\leq_I^{l\supseteq\wedge f}}$ induced by the logic $\mathcal{L}'_{\leq_I^{l\supseteq\wedge f}}$ is equivalent to the observational semantics defined by $\leq_I^{l\supseteq\wedge f}$.*

Proof. We just need to check that $\mathcal{L}'_{\leq_I^{l\supseteq\wedge f}} = \mathcal{L}'_{\leq_I^{l\supseteq}} \cup \mathcal{L}'_{\leq_I^{lf}}$, which is immediate. □

By replacing the I above by the generic N, we get the definitions and results for the general case.

The axiomatic characterization of the greatest lower bound of the readiness and failure trace semantics is much simpler: we simply put together the axioms for the orders defining both semantics.

Definition 8.7. The *meet semantics* $R \vee FT$ is that defined by the order $\sqsubseteq_{R\vee FT}$ generated by the set of axioms $\{B_1\text{–}B_4, (RS), (ND^R), (ND^{FT})\}$.

If we define $M_{R\vee FT}$ as $M_R \vee M_{FT}$, that is, $M_{R\vee FT}(x, y, w)$ holds if $I(x) \supseteq I(y)$ or $I(w) \subseteq I(y)$, we have the following characterization of $\sqsubseteq_{R\vee FT}$.

Proposition 8.8. *The order $\sqsubseteq_{R\vee FT}$ is that generated by the set of axioms $\{B_1\text{–}B_4, (RS), (ND^{R\vee FT})\}$, where $(ND^{R\vee FT})$ is the instantiation of the generic axiom (ND) with the condition $M_{R\vee FT}$.*

Proposition 8.9. *The semantics defined by the order $\sqsubseteq_{R\vee FT}$ is the finest semantics that is coarser than both the readiness and the failure trace semantics.*

Proof. Obvious since any semantics coarser than the readiness semantics has to satisfy $\{B_1\text{–}B_4, (RS), (ND^R)\}$, any one coarser than failure trace must satisfy $\{B_1\text{–}B_4, (RS), (ND^{FT})\}$, and $M_{R\vee FT}$ is equivalent to $M_R \vee M_{FT}$. □

Once again, the semantics defined by $\sqsubseteq^{R\vee FT}$ is not included in the ltbt spectrum and neither in our extended one; in particular, it is different from the failures semantics. To prove this fact we make essential use of the notion of *revival*, as defined by Reed, Roscoe, and Sinclair [46]. Revivals are sequences $a_1, \ldots, a_n(X, a)$ where $a_1, \ldots, a_n$ is a trace of the corresponding process after which the action a is offered, but the set of actions X is refused.

Proposition 8.10. *The meet semantics $R \vee FT$ is strictly finer than failure semantics.*

Proof. The inclusion $\sqsubseteq^{R\vee FT} \subsetneq \sqsubseteq^F$ is obvious since failures semantics is coarser than both the readiness and the failure trace semantics. To show that the inclusion is strict, note that any two processes related by $\sqsubseteq^{R\vee FT}$ do not only have the same failures but also the same revivals. This is indeed the case since all the axioms $u \preceq v$ in $\{B_1\text{–}B_4, (RS), (ND^R), (ND^{FT})\}$

preserve the revivals, which means $\mathit{Revivals}(\sigma(u)) \subseteq \mathit{Revivals}(\sigma(v))$ for every ground substitution σ, and the revivals order is a precongruence for the operators in BCCSP. For instance, for (ND^{FT}) we need to prove that $\mathit{Revivals}(\sigma(a(x+y))) \subseteq \mathit{Revivals}(\sigma(ax)) \cup \mathit{Revivals}(\sigma(a(y+w)))$ whenever $I(\sigma(w)) \subseteq I(\sigma(x))$. It is clear that the only non-trivial case occurs when $a(X,b) \in \mathit{Revivals}(\sigma(a(x+y)))$; then we have $(X,b) \in \mathit{Revivals}(\sigma(x+y))$ so that $X \in \mathit{Failures}(\sigma(x)) \cap \mathit{Failures}(\sigma(y))$ and $b \in I(\sigma(X))$ or $b \in I(\sigma(y))$. In the first case, $a(X,b) \in \mathit{Revivals}(\sigma(ax))$ whereas, in the second, $X \in \mathit{Failures}(\sigma(x+y))$ and therefore $a(X,b) \in \mathit{Revivals}(\sigma(a(x+y)))$. The case for (ND^R) is simpler. Once we know that $\sqsubseteq^{R\vee FT}$ preserves the revivals we only need to observe that the revivals cannot be obtained from the failures of a process. In particular, we have $ab \sqsubseteq^F a + a(b+c)$, but $a(\{c\},b) \in \mathit{Revivals}(ab) \setminus \mathit{Revivals}(a+a(b+c))$. □

Next we present the characterization of the revivals semantics in terms of our observational framework.

Definition 8.11. We define the order $\leq_N^{l\supseteq\vee f}$ by

$$\mathcal{T} \leq_N^{l\supseteq\vee f} \mathcal{T}' \iff \begin{array}{l}\text{for all } X_0a_1\dots X_n \in \mathcal{T}\\ \text{there is } \{Y_0a_1Y_1\dots Y_n^j \mid j\in J\} \subseteq \mathcal{T}' \text{ such that } X_n = \bigcup_{j\in J} Y_n^j.\end{array}$$

Proposition 8.12. *For all $p, q \in BCCSP$, $\mathit{Revivals}(p) \subseteq \mathit{Revivals}(q)$ if and only if $LGO_I(p) \leq_I^{l\supseteq\vee f} LGO_I(q)$.*

Proof. Note that $\leq_I^{l\supseteq\vee f}$ can be equivalently defined as

$$\mathcal{T} \leq_I^{l\supseteq\vee f} \mathcal{T}' \iff \begin{array}{l}\text{for all } X_0a_1\dots X_n \in \mathcal{T} \text{ and for all } a \in X_n\\ \text{there is } Y_0a_1\dots Y_n \in \mathcal{T}' \text{ with } a \in Y_n \text{ and } Y_n \subseteq X_n.\end{array}$$

Now, since $a_1\dots a_n(X,a) \in \mathit{Revivals}(p)$ if and only if there exists $X_0a_1\dots X_n \in LGO_I(p)$ such that $a \in X_n$ and $X_n \cap X = \emptyset$, we obtain the desired characterization. □

Definition 8.13. Given $\mathcal{T} \subseteq LGO_N$, $\overline{\mathcal{T}}^{\supseteq\vee f}$ is defined as

$$\overline{\mathcal{T}}^{\supseteq\vee f} = \{X_0a_1\dots X_n \mid \text{there is } \{Y_0a_1\dots Y_n^j \mid j\in J\} \subseteq \mathcal{T} \text{ with } X_n = \bigcup_{j\in J} Y_n^j\}.$$

This clearly indicates that $\leq_I^{l\supseteq\vee f}$ is in between $\leq_I^{lf\supseteq}$, defining the failures semantics, and $\leq_I^{lf}$, defining readiness semantics. This is useful for the proof of the axiomatic characterization of the revivals semantics.

Theorem 8.14. *The revivals semantics defined by $\sqsubseteq_I^{l\supseteq\vee f}$ is axiomatized by $\{B_1\text{–}B_4, (RS), (ND^{R\vee FT})$*

Proof sketch. It is quite similar to that of Theorem 5.4 for the case of failures semantics and, hence, also similar to the characterization of that semantics by means of acceptance trees [30] (and where the closure of the set of offers with respect to both union and convex closure is a critical argument), and this is why we only sketch it. In connection to that, recall that the application of the particular case of (ND) corresponding to (ND^{FT}) allowed us to join arbitrary states after the same trace, while that corresponding to (ND^R) allowed us to obtain a common continuation after the same action at any state reachable by the same trace. All this can be done now using $(ND^{R\vee FT})$; however, we cannot add to an arbitrary state an action offered at another state reachable by the same trace since to do that we needed the unlimited strength of axiom (ND). □

Note that for the join semantics $R \wedge FT$ the logical approach was the most direct way of defining it, whereas its equational characterization needed more care. For the meet semantics $R \vee FT$, the situation is just the opposite. As we have seen, $R \vee FT$ is axiomatized by putting together the axioms for R and those for FT; in contrast, the logic characterizing $R \vee FT$ is obtained by cleverly selecting the common part of the logics characterizing both R and FT. If we had defined the logical semantics by considering all the formulas from HML that are preserved by each semantics, then we could take the intersection of these sets as the logical semantics of any meet semantics. Since we defined our logical semantics by considering only a "basis" that generates the corresponding full set, we cannot simply take their intersection.

Definition 8.15. We define the set of formulas $\mathcal{L}'_{\leq_I^{l\supseteq \vee f}}$ as that generated by the clauses:

- $\top \in \mathcal{L}'_{\leq_I^{l\supseteq \vee f}}$;
- if $\sigma, \sigma_j \in \mathcal{L}'_I$ for all $j \in J$ then $(\sigma \wedge \bigwedge_{j\in J} \neg\sigma_j \top) \in \mathcal{L}_{\leq_I^{l\supseteq \vee f}}$;
- if $\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq \vee f}}$ and $a \in Act$ then $a\varphi \in \mathcal{L}'_{\leq_I^{l\supseteq \vee f}}$.

Note that in the second clause of this definition we have relaxed the condition in the definition of $\mathcal{L}'_R$ by considering an arbitrary failure (that defined by the set J), but only a positive offer (the action appearing in σ). This is how the revivals semantics becomes slightly finer than the failures semantics.

Proposition 8.16. *The logical semantics $\sqsubseteq'_{\leq_I^{l\supseteq \vee f}}$ induced by the logic $\mathcal{L}'_{\leq_I^{l\supseteq \vee f}}$ is equivalent to the observational semantics defined by $\leq_I^{l\supseteq \vee f}$.*

Proof. In this case we have taken $\mathcal{L}'_{\leq_I^{l\supseteq \vee f}} = \mathcal{L}'_{\leq_I^{l\supseteq}} \cap \mathcal{L}'_{\leq_I^{lf}}$. Then, to prove that it defines $R \vee FT$ it is enough to check that $p \not\leq_I^{l\supseteq \vee f} q$ implies that there exists φ in $\mathcal{L}'_{\leq_I^{l\supseteq \vee f}}$ such that $p \models \varphi$ and $q \not\models \varphi$, which is almost immediate. □

Again, by replacing the I above by the generic N, we get the definitions and results for the general case.

We can generalize most of the results obtained for the refusal semantics when $N = I$ to any reasonable local observation function such as T or S, once we interpret $\subseteq$ as the corresponding order and $=$ as the induced equivalence. However, in order to define the adequate observational characterization of the revivals semantics for a local observation (or constraint) N, we should look for the adequate "elements" of the universe of observations. This leads us to traces when N is T, but it is not so clear how to define those "elements" for a non-extensional semantics such as that obtained when N is S.

Let us conclude this section with a look at the picture in Figure 15 showing the real structure of the full (bidimensional!) diamond, that should be included in all the upper levels of the extended ltbt spectrum.

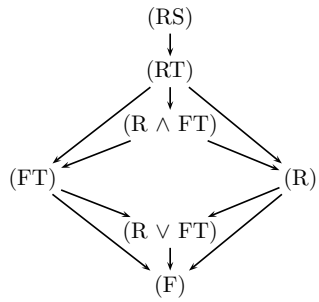


Figure 15: The real diamond below ready simulation.

9. Operational semantics

In this section we explain how to develop the semantics in the spectrum in an operational way. Certainly, this presentation could be argued to be ad-hoc at times since some "high-level" conditions are required in the SOS-like rules for some of the semantics. Moreover, the style of presentation at this Section is certainly less precise and detailed than in the previous ones. However, we believe it still provides some additional insight on the common properties of the semantics and also establishes a connection with our previous work on (bi)simulations up-to [23, 25] as a way to get coinductive characterizations of any "reasonable" process semantics.

Structural operational semantics was introduced by G. Plotkin in 1981, even though his seminal work was not published in a journal until 2004 [45]. In Section 2 we already presented a basic operational semantics for our processes as a starting point for the definition of all the semantics in the spectrum: a small-step semantics that collects the (atomic) actions executed by the processes into the corresponding transition system. By contrast, all the operational semantics in this section will be big-step semantics which directly return the adequate semantic values defining each of the semantics. They are generated by means of SOS-like rules that obtain these values in a compositional way. An extensive presentation of structural operational semantics covering all its variants can be found in [39].

9.1. **Local simulations up-to.** In order to characterize all the reasonable behavior preorders in a coinductive way we need to generalize constrained N-simulations (Definition 2.2) with N-simulations up-to an order $\sqsubseteq$.

Definition 9.1. Let $\sqsubseteq$ be a behavior preorder and N a relation over processes. We say that a binary relation S over processes is an *N-simulation up-to $\sqsubseteq$* if $S \subseteq N$ and S is a simulation up-to $\sqsubseteq$. Or equivalently, in a coinductive way, whenever we have pSq we also have:

- for every a, if $p \xrightarrow{a} p'_a$ then there exist q', q'_a such that $q \sqsupseteq q' \xrightarrow{a} q'_a$ and $p'_a S q'_a$;
- pNq.

We say that process p is N-simulated up-to $\sqsubseteq$ by process q, or that process q N-simulates process p up-to $\sqsubseteq$, written $p \stackrel{\sqsubseteq}{\sim}^{N}_{\sqsubseteq} q$, if there exists an N-simulation up-to $\sqsubseteq$, S, such that pSq.

We often just write $\stackrel{\sqsubseteq}{\sim}^{N}$, instead of $\stackrel{\sqsubseteq}{\sim}^{N}_{\sqsubseteq}$, when the behavior preorder is clear from the context.

We proved in [23] that all the preorders defining the semantics in the ltbt spectrum can be characterized as N-simulations up-to the corresponding equivalence relation $\equiv$, where N is the constraint defining the coarsest simulation semantics finer than the given semantics. For instance, the result for the semantics between failures semantics and ready simulation was the following.

Theorem 9.2 ([23])**.** *For every behavior preorder $\sqsubseteq$ satisfying the axiom (RS) and $\sqsubseteq \subseteq I$, we have $p \sqsubseteq q$ if and only if $p \stackrel{\sqsubseteq}{\sim}^{I}_{\sqsubseteq} q$.*

Table 6 shows the constraints defining the adequate constrained simulation order finer than each of the semantics in the linear time-branching time spectrum. Obviously, they coincide with the layer of the extended spectrum at which each semantics appear.

	T	S	CT	CS	F	R	FT	RT	PW	RS	PF	2N
C_O	U	U	C	C	I	I	I	I	I	I	V	W

$$pUq \iff true \qquad pVq \iff p \equiv_T q$$
$$pCq \iff (p = \mathbf{0} \text{ iff } q = \mathbf{0}) \qquad pWq \iff p \equiv_S q$$
$$pIq \iff I(p) = I(q)$$

Table 6: Constraints for the semantics in the ltbt spectrum.

Note that Theorem 9.2 is more subtle that it could appear: it characterizes a given preorder with a constrained simulation upto the preorder itself (Definition 9.1). Therefore, there are several semantics that share the same constraint. This characterization is indeed rather technical and the key point is that it allows to express any behavior preorder in a simulation-like fashion. We have used this characterization to prove many useful statements in our previous work[5] and we will use it again several times in the current Section.

In our proof of the completeness of the axiomatizations for the linear semantics in the spectrum in Section 5 we used a notion of normal form which, roughly, was defined by applying repeatedly to any term p the axiom ($ND_{\equiv}$) from right to left, for as long as possible. Propositions 5.2 and 5.3 were then the key results to complete the proof, and also lie behind the intuition for introducing now the notion of *local I-simulation up-to*.

Definition 9.3. For $Z \in \{F, R, FT, RT\}$ and $p = \sum_a \sum_i ap^i_a$, whenever we have a pair of indices i, j and a decomposition $p^j_a = r^j_a + s^j_a$ with $M_Z(p^j_a, r^j_a, s^j_a)$ we say that p is 1-locally Z-equivalent to $q = p + a(p^i_a + r^j_a)$, and we write $p \equiv^{l1}_Z q$. We say that p and q are locally Z-equivalent when they are related by the reflexive and transitive closure of $\equiv^{l1}_Z$, and then we write $p \equiv^{l}_Z q$.

For $Z \in \{F, R, FT, RT\}$ we refer to the I-simulations up-to $\equiv^{l}_Z$ as local I-simulations up-to $\equiv_Z$. We say that process p is locally I-simulated up-to $\equiv_Z$ by process q, or that

[5]For instance in [24] (Theorem 10) we provided an axiomatization for any behavior preorder starting from the equations of the corresponding equivalence.

process q locally I-simulates process p up-to $\equiv_Z$, written $p \sqsubseteq^I_{\sim_{\equiv^l_Z}} q$, if there exists a local I-simulation up-to $\equiv_Z$, S, such that pSq.

Local I-simulations up-to are enough to characterize the linear semantics in $\{F, R, FT, RT\}$. Note that we cannot get a local notion of bisimulation up-to equivalent to our unrestricted notion of bisimulation up-to.

Proposition 9.4. *For $Z \in \{F, R, FT, RT\}$ we have $p \sqsubseteq_Z q$ if and only if $p \sqsubseteq^I_{\sim_{\equiv^l_Z}} q$.*

Proof. The implication from right to left is an immediate consequence of Theorem 9.2. For the other, note that $\{(p, q) \mid p \sqsubseteq_Z q\}$ is a local I-simulation up-to $\equiv_Z$. Indeed, for any $p \xrightarrow{a} p^i_a$ we have $q \equiv^l_Z hnf^Z(q)$ and taking $hnf^Z(q) = \sum_a \sum_i a h^j_a$ there exists some j such that $hnf^Z(q) \xrightarrow{a} h^j_a$ and $p^i_a \sqsubseteq_Z h^j_a$. □

Example 9.5. Let us consider the processes $p = abc + abd$ and $q = a(bc + bd)$. We have $p \equiv_F q$ and we can check that $p \sqsubseteq^I_{\sim_{\equiv^l_F}} q$ since $p \sqsubseteq_{RS} q$. In order to prove that we also have $q \sqsubseteq^I_{\sim_{\equiv^l_F}} p$, we apply $\equiv^l_F$ to p to obtain $p \equiv^l_F p + q$ and then we obtain $q \sqsubseteq_{RS} p$.

By contrast, if we wanted to apply our bisimulation up-to characterization to prove directly that $p \equiv_F q$ then we would have to turn q into $q + p$ in order to simulate the transition $p \xrightarrow{a} bc$. This would correspond to the local application of $(ND^F_{\equiv})$ combined with that of

$$(RS_{\equiv}) \quad I(x) = I(y) \Longrightarrow a(x + y) \simeq a(x + y) + ax.$$

But if we replace the action a by a larger prefix $a_1 \dots a_n$ then we should also modify the process $q' = a_1 \dots a_n(bc + bd)$ in a non-local way in order to obtain $q'' = q' + p'$, so that we could suitably simulate the transition $p' = a_1 \dots a_n bc + a_1 \dots a_n bd \xrightarrow{a_1} a_2 \dots a_n bc$. Certainly, this is not necessary when checking $p' \equiv_F q'$ by means of local simulations up-to.

The coinductive characterization of the semantics by means of simulations up-to has at least two important advantages over that of using bisimulations up-to. First, we can characterize the orders defining the semantics and not just the induced equivalences; and second, we can use a local variant of the up-to mechanism so that we only need to rely on the equivalence relation $\equiv^l_Z$ for the up-to part.

9.2. **Operational rules for the linear semantics of processes.** In Section 9.1 we have introduced and proved some results that establish the framework using which we achieve our goal: to define for each of the classic linear semantics an operational semantics over BCCSP terms in such a way that we can use constrained simulations to characterize the considered semantics. For instance, if we consider the case of the failures preorder $\sqsubseteq_F$, we are going to define a new operational semantics for BCCSP terms $(\mathcal{P}, Act, \Rightarrow_F)$ such that $p \sqsubseteq_F q$ if and only if q ready simulates p in $(\mathcal{P}, Act, \Rightarrow_F)$.

Next we will concentrate first on the diamond of linear semantics coarser than ready simulation. All these semantics are based on the observation of the initial set of actions of each process, that can be obtained by application of the SOS-like rules in Figure 16.

The rules in Figure 17 define the transition relation $\Longrightarrow_Z$ that induces the operational semantics to characterize each of the Z-semantics. The transition relation $\longleftrightarrow_Z$ is an auxiliary relation that captures the iterated application of the axiom $(ND^Z_{\equiv})$. Rules (RF) and (TR) define reflexivity and transitivity of the relation $\longleftrightarrow_Z$. Finally, the rule (CL)

$$\mathbf{0} \longrightarrow_I \emptyset \qquad ap \longrightarrow_I \{a\} \qquad \frac{p \longrightarrow_I A \quad q \longrightarrow_I B}{p + q \longrightarrow_I A \cup B}$$

Figure 16: Rules that compute the set of initial actions of a process.

$$\text{(ND)}\ \frac{p \longrightarrow_I A_p \quad q \longrightarrow_I A_q \quad r \longrightarrow_I A_r \quad M_Z(A_p, A_q, A_r)}{ap + a(q + r) + s \longleftrightarrow_Z ap + a(q + r) + a(p + q) + s}$$

$$\text{(RF)}\ p \longleftrightarrow_Z p \qquad \text{(TR)}\ \frac{p \longleftrightarrow_Z q \quad q \longleftrightarrow_Z r}{p \longleftrightarrow_Z r}$$

$$\text{(CL)}\ \frac{p \longleftrightarrow_Z p' \quad p' \xrightarrow{a} q}{p \overset{a}{\Longrightarrow}_Z q}$$

Figure 17: Operational semantics characterizing the linear semantics.

combines the auxiliary relation $\longleftrightarrow_Z$ and the original operational transition relation $\longrightarrow$ (see Figure 2), to define the new labeled transitions $\Longrightarrow_Z$.

Definition 9.6. For $Z \in \{F, R, FT, RT\}$, the operational semantics for BCCSP terms is given by the labeled transition system $(\mathcal{P}, Act, \Longrightarrow_Z)$ where the transition relation $\Longrightarrow_Z$ is defined by the rules in Figure 17.

By abuse of notation, we have written $M_Z(A_p, A_q, A_r)$ to express that we check $M_Z(p, q, r)$ using the initials computed by $\longrightarrow_I$.

The relation $\Longrightarrow_Z$ has some interesting properties. First, it is an extension of the original transition system.

Proposition 9.7. *For $Z \in \{F, R, FT, RT\}$, p and q BCCSP processes, and α a sequence of actions in Act, we have that $p \overset{\alpha}{\Longrightarrow} q$ implies $p \overset{\alpha}{\Longrightarrow}_Z q$.*

Although usually some new transitions appear, the set of initial actions of any process always remains the same.

Corollary 9.8. *For $Z \in \{F, R, FT, RT\}$ and for any BCCSP process p, we have $I^{\rightarrow}(p) = I^{\rightarrow_Z}(p)$.*

It is also clear that, for any $Z \in \{F, R, FT, RT\}$, the auxiliary relation $\longleftrightarrow_Z$ preserves the equivalence $\equiv_Z$ because the rule (ND) corresponds to the application of axiom $(I\text{-}ND^Z_{\equiv})$, which is sound with respect to $\equiv^I_Z$.

Proposition 9.9. *For $Z \in \{F, R, FT, RT\}$ and any two BCCSP processes p and q, we have $p \longleftrightarrow_Z q$ implies $p \equiv_Z q$.*

Now we prove the main theorem in this section, that asserts that for each of the semantics in the considered diamond we can define the corresponding operational semantics as stated in Figure 17.

Theorem 9.10. *For $Z \in \{F, R, FT, RT\}$ and any two BCCSP processes p and q, we have*

$$p \sqsubseteq_Z q \iff p \sqsubseteq^{\Rightarrow_Z}_{RS} q.$$

Proof. We will apply our characterization of the orders $\sqsubseteq_Z$ by means of local I-simulations up-to at Proposition 9.4 to show that $p \sqsubseteq_{RS}^{\Rightarrow_Z} q$ implies $p \sqsubseteq^I_{\sim_{\equiv^l_Z}} q$. This is because any ready simulation over the transition system $\Longrightarrow_Z$ is also a local I-simulation up-to $\equiv_Z$. Indeed, if R is a ready simulation over the transition system $\Longrightarrow_Z$, and pRq, then whenever we have $p \xrightarrow{a} p'$ we also have $p \overset{a}{\Longrightarrow}_Z p'$, and therefore there is some $q \overset{a}{\Longrightarrow}_Z q'$ with $p'Rq'$. By definition of the transition system $\Longrightarrow_Z$, there is some process q'' such that $q \longleftrightarrow_Z q''$ and $q'' \xrightarrow{a} q'$. Then we also have $q \equiv^l_Z q''$, and thus R is indeed a local I-simulation up-to $\equiv_Z$.

To prove that $p \sqsubseteq^I_{\sim_{\equiv^l_Z}} q$ implies $p \sqsubseteq_{RS}^{\Rightarrow_Z} q$, we will check that the relation $\sqsubseteq^N_{\sim_{\equiv^l_Z}}$ is a ready simulation over the transition relation $\Longrightarrow_Z$. If $p \sqsubseteq^N_{\sim_{\equiv^l_Z}} q$, whenever $p \overset{a}{\Longrightarrow}_Z p'$ we have some process p'' such that $p \longleftrightarrow_Z p''$ and $p'' \xrightarrow{a} p'$. Then we also have $p \equiv_Z p''$, and so $p'' \sqsubseteq^N_{\sim_{\equiv^l_Z}} q$. From $p'' \xrightarrow{a} p'$ we now obtain that there are processes q' and q'' such that $q \equiv^l_Z q''$, $q'' \xrightarrow{a} q'$, and therefore we also have $q \longleftrightarrow_Z q''$, thus concluding the proof. □

As a consequence of our negative results at the end of Section 9.1, it is not possible to obtain an operational semantics locally defined from that which characterizes the linear semantics by means of bisimilarity. However, this can be done if we use mutual similarity instead of bisimulation.

Certainly, the fact that the characterizations in terms of bisimilarity cannot be defined in a local way is related to the fact that the transition systems generated by application of the algorithm in [17] are larger than those generated by our local transformation here. Unfortunately, it is true that our presentation does not magically lead (at least at the theoretical level) to more efficient algorithms to decide the equivalences with respect to the linear semantics (which are known to be quite hard to decide). Obviously, this is related to the fact that simulation is harder than bisimulation [37]. Even so, these are just theoretical worst case bounds, and it is nice to know that in practice we can apply a local transformation to generate the transition systems characterizing those semantics by means of the simulation orders, that in many concrete cases will not be too difficult to decide.

9.3. **Characterizing the semantics corresponding to other constraints.** Let us start by considering the case of the universal constraint U. As discussed in Section 3.2, if we use U in the condition M_Z it is clear that all the semantics in the corresponding diamond collapse into a single one: trace semantics. It is immediate to realize that the transition system to characterize it in terms of plain simulations is the same transition system $\Longrightarrow_F$ that we use to characterize the failures semantics by means of ready simulations.

Theorem 9.11. *The trace preorder $\sqsubseteq_T$ coincides with the simulation order on the transition system $\Longrightarrow_F$, that is, $p \sqsubseteq_T q$ iff $p \sqsubseteq_S^{\Rightarrow_F} q$.*

Even if this coincidence is a simple fact that reflects the relation between traces and failures semantics, it contributes to clarify it. In plain words, failures semantics is just traces semantics enriched by the observation of initials, so that the plain simulation order that implies the trace order becomes the ready simulation order.

For other, finer observers such as T we can also characterize the corresponding semantic orders, such as possible and impossible futures, in terms of local simulations up-to. We can use that result to justify that the corresponding transition systems $\Longrightarrow^T_Z$ would characterize the semantic orders $\sqsubseteq^T_Z$ in terms of T-simulations that preserve the set of traces of the

simulated process. In this case the corresponding operational characterization has to include rules for the computation of the set of traces $T(p)$ and this cannot certainly be done for infinite processes. But out of the computation of these sets, the rest of the rules for the generation of the corresponding transition systems $\Longrightarrow_Z^T$ are also valid, and their local character is still present.

9.4. **Application: trace deterministic normal forms.** As a simple application we present the example used by Klin in [36], that we already used in [22] to illustrate our coinductive characterization of the behavior preorders by means of our bisimulations up-to.

Definition 9.12. For any process $p = \sum_a \sum_i a p_a^i$ the *deterministic form* of p is defined as $Det(p) = \sum_a a Det(\sum_i p_a^i)$.

We wish to prove that p and $Det(p)$ are trace equivalent. We will do it by proving that they are simulation equivalent over the transition system $\Longrightarrow_F$.

Proposition 9.13. *For any process p we have $p \sqsubseteq_F Det(p)$.*

Proof. We will prove that $p \sqsubseteq_S^{\Rightarrow_F} Det(p)$ by showing that $R = \{(p, Det(p+q)) \mid p, q \text{ processes}\}$ is a simulation for the transition system $\Longrightarrow_F$. For $q = \sum_a \sum_j a q_a^j$ we have $Det(p+q) = \sum_a Det(\sum_i p_a^i + \sum_j q_a^j)$. Then, for any $p \stackrel{a}{\Longrightarrow}_F p'$ we have $p = p_a^i + \sum_k r_a^k$, for some index i and $p_a^k = r_a^k + s_a^k$ a decomposition of any of the rest of the summands of p. We have $Det(p+q) \stackrel{a}{\longrightarrow} Det(\sum_i a p_a^i + \sum_j a q_a^j) = Det((p_a^i + \sum_k r_a^k) + (\sum_k r_a^k + \sum_j q_a^j))$, so that we also have $Det(p+q) \stackrel{a}{\Longrightarrow}_F Det((p_a^i + \sum_k r_a^k) + (\sum_k r_a^k + \sum_j q_a^j))$, with $(p_a^i + \sum_k r_a^k, Det((p_a^i + \sum_k r_a^k) + (\sum_k r_a^k + \sum_j q_a^j))) \in R$. □

Proposition 9.14. *For any process p we have $Det(p) \sqsubseteq_F p$.*

Proof. We will prove that $Det(p) \sqsubseteq_S^{\Rightarrow_F} p$ by showing that $R = \{(Det(p), p)\}$ is a simulation for the transition system $\Longrightarrow_F$. Since $Det(p)$ is deterministic for each $a \in Act$ there is a unique transition $Det(p) \stackrel{a}{\Longrightarrow}_F Det(\sum_i p_a^i)$. By applying the definition of $\stackrel{a}{\Longrightarrow}_F$ we have $p \stackrel{a}{\Longrightarrow}_F \sum_i p_a^i$, and clearly we have $(Det(\sum_i p_a^i), \sum_i p_a^i) \in R$. □

Although this is a very simple example, it is interesting to compare the proof above with that in [22]. This proof is simpler and more natural, mainly because the proof obligations to check bisimulations forced us to remove the sub-terms that were not in the chosen transition when we had to simulate it. This is not necessary for any of the two simulations that are needed to check mutual simulation, as done above. Obviously, this is also related to the impossibility to obtain a notion of local bisimulation up-to characterizing the equivalence under any of the linear semantics.

10. Conclusions and some future work

Throughout this paper we have provided a global outline of process semantics from different points of view, each of which reveals some of the key ingredients for a more uniform comprehension of those semantics. We have noted that the family consisting of the simulation semantics—constrained simulations, in its generalized version—plays an essential role in the class of process semantics, becoming the cornerstone for sorting and classifying the remaining semantics.

From a framework in which, based on observational trees, denotational semantics are assigned —Section 4—we have been able to prove that the spectrum of process semantics can be structured by means of layers that are induced by the simulations. Each layer is dominated by a simulation semantics that determines the finest distinction available for that layer. The remaining semantic families are also described by abstracting or simplifying the observations needed for the corresponding layer. In particular, below each constrained simulation there appear the corresponding versions for each of the classic linear semantics—failures, readiness, failure trace, and ready trace—and, as we saw in Section 8, other semantics are also explained within our framework.

This observational characterization allowed us to offer a new insight into the axiomatic characterization of the semantics—Sections 3 and 5—revealing a uniformity lacking in all previous studies. To characterize any of the orders that define a process semantics, we have proved that it is enough to use two parametric axioms: one of the required axioms is that for the generalized simulation of the corresponding layer while the other, when it is present, has to do with the reduction of non-determinism that is carried out in each semantics.

Analogously, in Sections 6 and 7 we showed how to characterize process semantics by means of sets of Hennesy-Milner logic formulas out of their observational characterization, and finally we have also discussed a unified operational presentation of the semantics in the extended spectrum.

One of the more obvious lines for future work would be to consider those semantics that allow for an inner, non-visible action, known as *weak semantics*. Actually, some promising results have already been obtained that make clear the regularity and generality present in the domain of weak semantics. In particular, in [16] it is proved that it is possible to apply to weak semantics the algorithm to obtain axiomatic characterizations of semantic equivalences from the axioms for corresponding order [21]. And [2, 3] provides a detailed study of the axiomatization of weak simulation semantics.

Let us also cite here the recent work by Anti Valmari [53], where he presents the full catalogue of (weak) linear-time congruences for finite state systems. Certainly, it is interesting to limit somehow the class of "reasonable" semantics for processes, but this has not been so much the intention of our work in this paper. In fact, it is interesting to note that the results in the paper referenced above limit the set of semantics to explore in a quite personal way: for instance, the semantics of failure traces and that of ready traces are not included in the category, because Valmari (implicitly) considers that they are not "linear-time enough".

Another interesting approach consists in the use of coalgebras—following the work, among others, of Jesse Hughes and Bart Jacobs [35]—where powerful categorical techniques allow to connect the idea of simulation with that of bisimulation, which is central in the coalgebraic setting. More concretely, these techniques were successfully used in [26] to relate classic and probabilistic bisimulation.

References

[1] Luca Aceto. Some of my favourite results in classic process algebra. In *In Bulletin of the EATCS*, pages 89–108, 2003.

[2] Luca Aceto, David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Anna Ingólfsdóttir. Axiomatizing weak ready simulation semantics over bccsp. In Cerone and Pihlajasaari [15], pages 7–24.

[3] Luca Aceto, David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Anna Ingólfsdóttir. The equational theory of weak complete simulation semantics over bccsp. In Bieliková et al. [12], pages 141–152.

[4] Luca Aceto, Wan Fokkink, and Anna Ingólfsdóttir. Ready to preorder: get your BCCSP axiomatization for free! In *Algebra and Coalgebra in Computer Science, Second International Conference, CALCO 2007*, volume 4624 of *Lecture Notes in Computer Science*, pages 65–79. Springer, 2007.

[5] Luca Aceto, Wan Fokkink, Anna Ingólfsdóttir, and Bas Luttik. Finite equational bases in process algebra: Results and open questions. In *Processes, Terms and Cycles*, volume 3838 of *Lecture Notes in Computer Science*, pages 338–367. Springer, 2005.

[6] Luca Aceto, Anna Ingólfsdóttir, Kim G. Larsen, and Jiri Srba. *Reactive Systems: Modelling, Specification and Verification*. Cambridge University Press, 2007.

[7] Jos C. M. Baeten. A brief history of process algebra. *Theoretical Compututer Science*, 335(2-3):131–146, 2005.

[8] Jos C. M. Baeten, Twan Basten, and Michel A. Reniers. *Process Algebra: Equational Theories of Communicating Processes (Cambridge Tracts in Theoretical Computer Science)*. Cambridge University Press, 2009.

[9] Jos C. M. Baeten, Jan A. Bergstra, and Jan Willem Klop. Ready-trace semantics for concrete process algebra with the priority operator. *The Computer Journal*, 30(6):498–506, 1987.

[10] Jos C.M. Baeten and W.P. Weijland. *Process Algebra*. Cambridge Tracts in Computer Science. Cambridge University Press, 1990.

[11] Jan A. Bergstra and Jan Willem Klop. Process algebra for synchronous communication. *Information and Control*, 60(1-3):109–137, 1984.

[12] Mária Bieliková, Gerhard Friedrich, Georg Gottlob, Stefan Katzenbeisser, and György Turán, editors. *SOFSEM 2012: Theory and Practice of Computer Science - 38th Conference on Current Trends in Theory and Practice of Computer Science, pindlerv Mlýn, Czech Republic, January 21-27, 2012. Proceedings*, volume 7147 of *Lecture Notes in Computer Science*. Springer, 2012.

[13] Bard Bloom, Sorin Istrail, and Albert R. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42(1):232–268, 1995.

[14] Stephen D. Brookes, C.A.R. Hoare, and A. William Roscoe. A theory of communicating sequential processes. *Journal of the ACM*, 31(3):560–599, 1984.

[15] Antonio Cerone and Pekka Pihlajasaari, editors. *Theoretical Aspects of Computing - ICTAC 2011 - 8th International Colloquium, Johannesburg, South Africa, August 31 - September 2, 2011. Proceedings*, volume 6916 of *Lecture Notes in Computer Science*. Springer, 2011.

[16] Taolue Chen, Wan Fokkink, and Rob J. van Glabbeek. Ready to preorder: The case of weak process semantics. *Information Processing Letters*, 109(2):104–111, 2008.

[17] Rance Cleaveland and Matthew Hennessy. Testing equivalence as a bisimulation equivalence. *Formal Asp. Comput.*, 5(1):1–20, 1993.

[18] David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino. Coinductive characterisations reveal nice relations between preorders and equivalences. In *First International Conference on Foundations of Informatics, Computing and Software (FICS 2008)*, volume 212 of *Electronic Notes in Theoretical Computer Science*, pages 149–162. Elsevier, 2008.

[19] David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino. On the unification of process semantics: Equational semantics. *Electronic Notes in Theoretical Computer Science*, 249:243–267, 2009.

[20] David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino. On the unification of process semantics: Observational semantics. In *SOFSEM 2009: Theory and Practice of Computer Science, 35th Conference on Current Trends in Theory and Practice of Computer Science*, volume 5404 of *Lecture Notes in Computer Science*, pages 279–290. Springer, 2009.

[21] David de Frutos Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino. Ready to preorder: an algebraic and general proof. *Journal of Logic and Algebraic Programming*, 78(7):539–551, 2009. doi:10.1016/j.jlap.2008.09.001.

[22] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. Bisimulations up-to for the linear time-branching time spectrum. In *CONCUR 2005 - Concurrency Theory, 16th International Conference*, volume 3653 of *Lecture Notes in Computer Science*, pages 278–292. Springer, 2005.

[23] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. Simulations up-to and canonical preorders (extended abstract). In *Structural Operational Semantics SOS 2007*, volume 192 of *Electronic Notes in Theoretical Computer Science*, pages 13–28. Elsevier, 2007.

[24] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. Universal coinductive characterizations of process semantics. In *5th IFIP International Conference on Theoretical Computer Science*, volume 273 of *IFIP*, pages 397–412. Springer, 2008.
[25] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. (Bi)simulations up-to characterise process semantics. *Information and Computation*, 207(2):146–170, 2009.
[26] David de Frutos-Escrig, Miguel Palomino, and Ignacio Fábregas. Multiset bisimulations as a common framework for ordinary and probabilistic bisimulations. In *Formal Techniques for Networked and Distributed Systems - FORTE 2008, 28th IFIP WG 6.1 International Conference*, volume 5048 of *Lecture Notes in Computer Science*, pages 283–298. Springer, 2008.
[27] Cédric Fournet, Tony Hoare, Sriram K. Rajamani, and Jakob Rehof. Stuck-free conformance. In Rajeev Alur and Doron A. Peled, editors, *Computer Aided Verification. 16th International Conference, CAV 2004, Boston, MA, USA, July 13–17, 2004. Proceedings*, volume 3114 of *Lecture Notes in Computer Science*, pages 242–254. Springer, 2004.
[28] Per Brinch Hansen. *The Origins of Concurrent Programming: From Semaphores to Remote Procedure Calls*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.
[29] Matthew Hennessy. Acceptance trees. *Journal of the ACM*, 32(4):896–928, 1985.
[30] Matthew Hennessy. *Algebraic Theory of Processes*. MIT Press, 1988.
[31] Matthew Hennessy and Robin Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32:137–161, 1985.
[32] C.A.R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.
[33] Tony Hoare. *Process Algebra: A Unifying Approach*, volume 3525 of *Lecture Notes in Computer Science*, pages 36–60. Springer Berlin / Heidelberg, 2005.
[34] Tony Hoare and He Jifeng. *Unifying Theories of Programming*. Prentice Hall, 1998.
[35] Jesse Hughes and Bart Jacobs. Simulations in coalgebra. *Theoretical Computer Science*, 327(1–2):71–108, 2004.
[36] Bartek Klin. A coalgebraic approach to process equivalence and a coinductive principle for traces. In *CMCS'04: 7th International Workshop on Coalgebraic Methods in Computer Science*, volume 106 of *Electronic Notes in Theoretical Computer Science*, pages 201–218. Elsevier, 2004.
[37] Antonín Kucera and Richard Mayr. Why is simulation harder than bisimulation? In *CONCUR 2002 - Concurrency Theory, 13th International Conference, Proceedings*, volume 2421 of *Lecture Notes in Computer Science*, pages 594–610. Springer, 2002.
[38] Kim G. Larsen and Arne Skou. Bisimulation through probabilistic testing (preliminary report). In *Principles of Programming Languages, 16th ACM SIGACT-SIGPLAN Symposium – POPL '89*, pages 344–352. ACM Press, 1989.
[39] Chris Verhoef Luca Aceto, Wan Fokkink. *Handbook of Process Algebra*, chapter 3, Structural operational semantics, pages 197–292. Elsevier, 2001.
[40] Gerald Lüttgen and Walter Vogler. Ready simulation for concurrency: It's logical! *Information and Computation*, 208(7):845–867, 2010.
[41] Robin Milner. *A Calculus of Communicating Systems*. LNCS 92. Springer, 1980.
[42] Robin Milner. *Communication and Concurrency*. Prentice Hall, 1989.
[43] Ernst R. Olderog and C.A.R. Hoare. Specification-oriented semantics for communicating processes. *Acta Informatica*, 23(1):9–66, 1986.
[44] Iain Phillips. Refusal testing. *Theoretical Computer Science*, 50(3):241–284, 1987.
[45] Gordon D. Plotkin. A structural approach to operational semantics. *Journal of Logic and Algebraic Programming*, 60-61:17–139, 2004.
[46] Joy N. Reed, A. William Roscoe, and Jane E. Sinclair. Responsiveness and stable revivals. *Formal Aspects of Computing*, 19:303–319, 2007.
[47] David Romero-Hernández and David de Frutos-Escrig. On the unification of process semantics: Logical semantics. In *Structural Operational Semantics, SOS'11*, volume 62 of *EPTCS*, pages 47–61, 2011.
[48] A. William Roscoe. Revivals, stuckness and the hierarchy of CSP models. *Journal of Logic and Algebraic Programming*, 78(3):163–190, 2009.
[49] A. William Roscoe. *Understanding Concurrent Systems (Texts in Computer Science)*. Springer, 2010.
[50] Davide Sangiorgi. On the origins of bisimulation and coinduction. *ACM Trans. Program. Lang. Syst.*, 31(4):41 pages, 2009.
[51] Davide Sangiorgi. *Introduction to Bisimulation and Coinduction*. Cambridge University Press, 2012.

[52] Dana Scott and Christopher Strachey. Towards a mathematical semantics for computer languages. Programming Research Group Technical Monograph PRG-6, Oxford Univ. Computing Lab., 1971.
[53] Anti Valmari. All linear-time congruences for finite ltss and familiar operators. In *Application of Concurrency to System Design (ACSD), 2012 12th International Conference on*, pages 12–21. IEEE, 2012.
[54] Rob J. van Glabbeek. *Comparative Concurrency Semantics and Refinement of Actions*. PhD thesis, Free University, Amsterdam, 1990. Second edition available as *CWI tract* 109, CWI, Amsterdam 1996.
[55] Rob J. van Glabbeek. The linear time-branching time spectrum. In *CONCUR '90 Theories of Concurrency: Unification and Extension*, number 458 in Lecture Notes in Computer Science, pages 278–297. Springer-Verlag, 1990.
[56] Rob J. van Glabbeek. The linear time - branching time spectrum II. In *CONCUR '93 - Concurrency Theory, 5th International Conference*, volume 715 of *Lecture Notes in Computer Science*, pages 66–81. Springer, 1993.
[57] Rob J. van Glabbeek. Notes on the methodology of ccs and csp. *Theoretical Computer Science*, 177(2):329–349, 1997.
[58] Rob J. van Glabbeek. *Handbook of Process Algebra*, chapter 1, The Linear Time – Branching Time Spectrum I: The Semantics of Concrete, Sequential Processes, pages 3–99. Elsevier, 2001.
[59] Marc Voorhoeve and Sjouke Mauw. Impossible futures and determinism. *Information Processing Letters*, 80(1):51–58, 2001.

Chapter 6

Distances between finite processes

"... this piece we have now before us, which this worthy gentleman has in his hands, not only is no barber's basin, but is as far from being one as white is from black, and truth from falsehood..."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter-45 Part-1

"... esta pieza que está aquí delante y que este buen señor tiene en las manos no sólo no es bacía de barbero, pero está tan lejos de serlo como está lejos lo blanco de lo negro y la verdad de la mentira..."

At the same time that we were concluding our logical unification theory presented in Chapter 5, we started to consider what more we could say when two processes are not equivalent. In this sense, the notion of distance between processes appeared as a new research direction. Our objective was to measure "how much not equivalent" are two processes with respect to the reference semantics. We presented a metric version of the equivalences for processes for a simple process language that includes prefixing, choice and variables.

6.1 Our new proposal for a distance between processes

The paper presented in this section shows the first operational characterization of our distance. This notion captured the cost of the necessary changes to achieve two equivalent processes, starting from the two compared ones. We started by presenting the operational rules for the case of bisimulation and simulation distances. After that, the generality of our approach capacitated us to obtain a (reasonable) distance for each of the semantic in the *ltbt-spectrum*.

I personally presented the following paper in the 32^{nd} *IFIP International Conference on Formal Techniques for Networked and Distributed Systems (FMOODS / FORTE 2012)* held in Stockholm (Sweden). That year, the program committee included, among others names, those of Grigore Rosu, Doron Peled and Uwe Nestmann. Our paper has already been cited in several publications of high quality, as can be checked for instance by using *Google Scholar*.

Defining Distances for All Process Semantics*

David Romero Hernández and David de Frutos Escrig

Dpto. Sistemas Informáticos y Computación
Facultad CC. Matemáticas, Universidad Complutense de Madrid, Spain
dromeroh@pdi.ucm.es, defrutos@sip.ucm.es

Abstract. Recently several authors have proposed some notions of distance between processes that try to quantify "how far away" is a process to be related with some other with respect to a certain semantics. These proposals are usually based on the simulation game, and therefore are mainly defined for simulation semantics or other semantics more or less close to these. These distances have a local character since only one of the successors of each state is taken into account in their computation. Here, we present an alternative proposal exploiting the fact that processes are trees. We define the distance between two of them as the cost of the transformations that we need to apply to get two processes related by the corresponding semantics. Our new distances can be uniformly defined for all the semantics in the ltbt-spectrum.

1 Introduction and Motivation

We are thirsty, but we hate those boring machines that only offer a few products. But we are very happy with the machine at our institution that offers a wide variety of beverages. So, each day we can go to the machine with our selected chosen item and get our bottle. But if some day the machine is out of that, then we have to choose another drink, and that day we are not so happy... Certainly, if it is only a single kind of drink that is missing we will probably stay very happy, but if something happens and the machine today offers only a single beverage, then we will be probably not so happy...

We have a collection of items in some numbered "collector desk". We look for a product by reminding its assigned number. But today, for some reason, somebody has interchanged two items and then if we look for one of them we will find the other, and we will have to make our job using it, obviously not so well as if we had found the desired item. But if one day the desk collapses and somebody has to put the items in the places without knowing their places, and he is wrong in all the cases, then for sure we will fail when looking for any of the items.

There is a lottery in the club and everybody expects that all the balls corresponding to the sold tickets will be in the bag. But for some reason the set of balls does not exactly corresponds with that of sold tickets. Certainly, the raffle

* Partially supported by the Spanish projects TESIS (TIN2009-14312-C02-01), DESAFIOS10 (TIN2009-14599-C03-01) and PROMETIDOS S2009 / TIC-1465.

H. Giese and G. Rosu (Eds.): FMOODS/FORTE 2012, LNCS 7273, pp. 169–185, 2012.

is not fair, but how much unfair? An obvious reply will take into account the number of tickets that were not presented in the bag.

All these are simple "real life" situations, that we can easily model by means of a process with some kind of choice (either internal or external), where the number of choices in the initial model is large. This corresponds to the ideal situation, but if something is wrong, the choice is not the same and this would produce a process that does not fully satisfy our expectations. Then, we want to measure how far away we are from the desired behavior.

At the technical level, we want to define adequate distances between processes which measure, in a reasonable way, the gap between any behavior and the corresponding "expected" one. Of course, if we are talking about behaviors, then the first thing to fix is the reference semantics. There are plenty of proposals for process semantics, which have been presented in several versions of the linear-time branching-time (ltbt) spectrum [14, 5].

In the last few years we can find in the literature several proposals for distances between processes associated to a certain range of process semantics, but in all the cases far from being applicable to the whole spectrum [1]. Most of them, if not all, base their definitions on the (bi)simulation game that characterizes (bi)simulations between processes [11, 3, 2]. Although these are branching semantics, their co-inductive characterizations provide a (partially) local way to compare processes by considering, one by one, all the possible transitions from the compared states. The rules of these games state that any a-transition should be replicable by another a-transition of the other process; otherwise, we would have found a proof of non-bisimilarity (or that of non existence of a simulation) of the two compared processes.

Starting from them, the modified distance games allow the defender to reply an a-move by means of another b-move, where we could have $a \neq b$. Then he should pay to the attacker as the provided distance between these two actions, $d(b, a)$, states. Obviously, the attacker tries to maximize his profit by making his appropriate moves, while the defender tries to minimize them with his moves. Finally, the value of this game provides the (bi)simulation distance between the two compared processes w.r.t. the provided distance between actions, d.

Certainly, we could agree about the naturalness of these approaches, which in fact are proved to be correct, in the sense that the distance between two processes is 0, if and only if, they are (bi)similar. But, if we apply these distances to the formalizations of our three examples above, considering the discrete distance between processes (given by $d(a, a) = 0$ and $d(a, b) = 1$ if $a \neq b$) and taking p_n as the corresponding "ideal" behavior, where n is the desired number of choices, p_{n-1} the slightly "incorrect" approximation, and p_1 the poor approximation with a single choice, we obtain $d(p_n, p_{n-1}) = 1$, probably as expected, but a bit surprisingly, we also have $d(p_n, p_1) = 1$. In our opinion, it would be much more informative to get instead $d(p_n, p_1) = n - 1$, in such a way that if we consider the general approximation p_k of the ideal process p_n, which offers exactly k of the actions, then we have $d(p_n, p_k) = n - k$, and also $d(p_k, p_1) = k - 1$.

Why these known distances between processes fail to notice the quantity of choices that are lost? This is simple: just because the "local" character of the distance game. It certainly observes any of the lost actions, but this only happens at different plays of the game, each of them producing a profit $d(a_i, a_1) = 1$ to the attacker, so that the "final" profit (the value of the game, that generates $d(p_n, p_k)$) is always 1, when $k < n$, whatever the number of lost choices, $n - k$ was.

Even if we definitely advocate for a distance which will get $d(p_n, p_k) = n - k$, and in fact we will provide such a distance, we could still look for "justifications" of the distance produced by the game approaches: if we only study the computations of the processes "one by one" (certainly step by step, in order to get the characterizations of the branched semantics, instead of just the trace semantics) then we will never realize that several choices were lost at the same time (we only notice that "each one of them" was lost, but this is not enough).

What is the problem? (and then, how can we solve it?). Simulations define branched behaviors that are roughly trees which consider all computations of each process together [15]. These trees can be seen as "global" values (or full behaviors) of the process. Equality (resp. containment) of trees is defined (in a coalgebraic way) by bisimulation (resp. simulation), and then (in a partially local way) by the bisimulation (resp. simulation) game. We could say that this is the "magic" of (bi)simulation, but when we introduce distances between actions and we try to lift them up to the branched behaviors by means of the distance game, then we find that the obtained values are not able to capture the branched structure, because the value of the game is obtained by the minimax algorithm, which chooses the critical path generated by the application of the optimal strategies of both players, but is not able to "add" the differences observed at different branches. Indeed, we are using *max* instead of *add* when computing the value of the distance games, and then we cannot capture the "global" distances as required by the situations in our introductory examples.

As a matter of fact, the reason why the plain (bi)simulation game is able to capture a branched semantics is because we are interested in checking equality. This can be done by a boolean function which only considers boolean values, e.g. 0 for equal and 1 for unequal. Then, any move that the defender cannot match produces some 1, so the application of max would produce the value 1. But in this discrete domain, max can also be used to compute addition, which in fact coincides with disjunction. Instead, as soon as we have a more informative domain for the values of distances, then max and add become two different operations. It is clear that the first is only able to transmit a partial information about the branched behaviors, while addition collects all the "local" differences to compute a much more reasonable concept of global distance.

Once we have our mechanism to compute our global bisimulation distance, we will see that a quite simple customization, gives us a nice notion of distance for each of the semantics in the ltbt-spectrum. Roughly we just need to combine the preorder defining each of the other semantics in the ltbt-spectrum—see Fig.1—(or equivalently, the inequalities that are included in their axiomatizations), with the rules which produce the values of our bisimulation distance, .

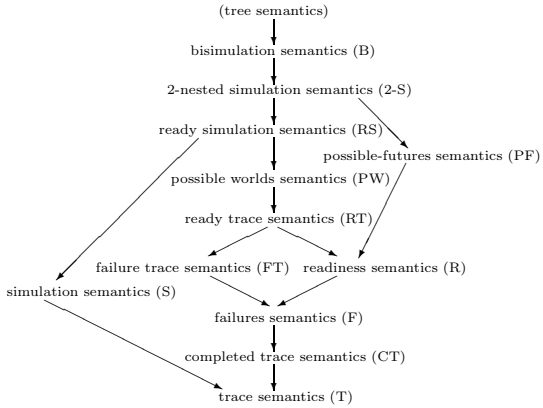


Fig. 1. The ltbt-spectrum

2 Preliminaries

All the semantics from the ltbt-spectrum [14, 5, 6] that we consider can be defined over arbitrary (possibly infinite) processes whose operational semantics is defined by means of a labelled transition system (lts) $\mathcal{P} = (Proc, Act, \rightarrow)$. We will use the classical notation $p \xrightarrow{a} p'$ to represent the transitions of processes. Moreover, it is also useful to have a syntactic notation for representing finite processes. We will use BCCSP [14, 5].

Definition 1. *Given a set of actions Act, the set BCCSP(Act) of processes is that defined by the BNF-grammar:* $p ::= \mathbf{0} \mid ap \mid p + q$. *The very well known operational semantics of BCCSP [14, 5] is defined by:*

$$(1)\ \frac{}{ap \xrightarrow{a} p} \qquad (2)\ \frac{p \xrightarrow{a} p'}{p + q \xrightarrow{a} p'} \qquad (3)\ \frac{q \xrightarrow{a} q'}{p + q \xrightarrow{a} q'}$$

In order to simplify the presentation, we start by considering a classic (symmetric) distance between actions $\overline{d} : Act \times Act \rightarrow \mathbb{N}$ with $\overline{d}(a, b) = \overline{d}(b, a)\ \forall a, b \in Act$. Let us recall that any distance has besides to satisfy the following two properties: $\overline{d}(a, b) = 0 \Leftrightarrow a = b$, $\overline{d}(a, c) + \overline{d}(c, b) \leq \overline{d}(a, b)\ \forall a, b, c \in Act$. Later, in Sect.6, we will discuss when an asymmetric quasi-distance could be used instead, and which is the intuitive meaning of the distances between processes that can be obtained using them.

We can represent any process as a tree (finite or infinite). Then a first approach to the definition of a distance measuring how far away is a process p of being equivalent to some other q, would study the differences between the trees which represent both processes, seeing what we have to change in order to turn them into two equivalent processes. Let us start by considering ordered trees, where we have a set of ordered sons for each node of the tree. We can present these trees as terms $\sum_{i=1}^{n} a_i p_i$, where $n = 0$ produces the empty tree $\mathbf{0}$.

Definition 2. *We say that an ordered tree p is at most at distance d from another tree q, w.r.t. the symmetric distance between actions $\overline{d}$, and then we write $d_{\overline{d}}(p,q) \leq d$, if and only if:*

- $d \geq 0$ *and* $p = q = \mathbf{0}$, *or*
- $p = \sum_{i=1}^{n} a_i p_i$, $q = \sum_{i=1}^{n} b_i q_i$, *and* $d = \sum_{i=1}^{n} d_i + \sum_{i=1}^{n} \overline{d}(a_i, b_i)$ *with* $d_{\overline{d}}(p_i, q_i) \leq d_i \ \forall i = 1 \ldots n$.

It is clear that this definition only produces (finite) distances between trees which have exactly the same structure. For instance, for the processes $p = a + b$ and $q = c + d$ we obtain $d_{\overline{d}}(p,q) \leq \overline{d}(a,c) + \overline{d}(b,d)$. However, if we want to compare $r = a$ and $s = b + c$, we will get no finite value d for which $d_{\overline{d}}(r,s) \leq d$, and then we could say that $d_{\overline{d}}(r,s) = \infty$.

Moreover, when comparing two infinite trees we will only obtain a finite distance if the number of disagreements between them is finite. Certainly, this will be the expected result if we simply add the cost of all these mismatches. But it is important to notice that the simple approach here proposed will never been able to compute distances between infinite trees with infinitely many mismatches. Therefore, in the following we will restrict ourselves to the case of finite processes, leaving the case of infinite processes for our conclusions.

It is also true that in this simple scenario when we compare two trees with the same structure, we could directly obtain the distance between them. But we preferred to introduce this indirect presentation using bounds, because this will be later needed when considering more complicated scenarios. Certainly, the order between the summands is important in ordered trees. As a consequence, if we consider $p' = b + a$ and $\overline{d}(a,b) = 1$, we obtain $d_{\overline{d}}(p,p') \leq 2$, and definitely not $d_{\overline{d}}(p,p') \leq 0$.

But trees representing processes are unordered: each node has attached a set of subtrees, and this even implies that no identical sons are allowed. In fact, this corresponds to considering processes "up-to" bisimulation. Then, in order to define a reasonable and well behaved notion of (bound of the) distance between processes, we apply a push-out of the definition above and that of bisimulation. So we get a rewriting procedure where we try to change any of the two compared processes into the other: Either changing one of the actions in a tree by other, but then we need to pay for it, as stated by the function $\overline{d}$; or we simply apply for free to any subtree of them any of the bisimulation axioms:

$$(B1)\ x + y \simeq y + x \qquad (B2)\ x + x \simeq x$$

$$(B3)\ (x + y) + z \simeq x + (y + z) \qquad (B4)\ z + \mathbf{0} \simeq z$$

Obviously, this procedure is non-deterministic and different possible applications lead us to several (different) "distances", and this is why we need to talk about "bounds" of the distance between p and q.

Definition 3. *We say that an unordered tree p is at most at distance d from another tree q, w.r.t. the symmetric distance between actions $\overline{d}$, and then we write $d_{\overline{d}}(p,q) \leq d$, if and only if:*

- *(C1) $p = ap'$, $q = bp'$, and $d \geq \overline{d}(a,b)$, or*
- *(C2) $p = p' + r$, $q = q' + r$, and $d \geq d_{\overline{d}}(p',q')$, or*
- *(C3) $p = ap'$, $q = aq'$, and $d \geq d_{\overline{d}}(p',q')$, or*
- *(C4) $d \geq 0$ and q can be obtained from p by application of (B1)-(B4), or*
- *(C5) There exist r, d' and d'' s.t. $d' \geq d_{\overline{d}}(p,r)$, $d'' \geq d_{\overline{d}}(r,q)$ and $d \geq d' + d''$.*

$(C1)$ corresponds to a single application of Def.1 producing a single change at the root of p. $(C2)$ and $(C3)$ allow the contextual application of $(C1)$ at any place, thus generating the possibility to change any action a in p by any other action b, paying $\overline{d}(a,b)$ for it. $(C4)$ introduces the possibility of transforming any process p into another bisimilar q, for free. Finally, $(C5)$ tells us that by adding the costs of the steps of any transformation that produces q from p, we obtain an upper bound of the distance between p and q.

We could obtain "the" distance between two trees by considering the minimal value d for which we have $d_{\overline{d}}(p,q) \leq d$. But unfortunately this corresponds to a global study of the set of derivations that produces the bounds. We prefer to avoid the explicit consideration of those "exact" distances, since it seems not possible to introduce the computation of these minimal values in our approach in a manageable way.

Moreover, it is easy to see that these distances would correspond to the shortest path in the graph whose nodes are processes, and the valued arcs correspond to the cost of the basic allowed transformations between them induced by rules $(C1) - (C4)$; $(C5)$ states somehow the Bellman's optimality principle. As a consequence we do not need "all the strength" of rule $(C5)$ which allows us to compose a path by concatenating two arbitrary paths, but it certainly includes the (needed) case in which the first path is a single step. However, by including this general rule we obtain a more symmetric definition, where those single steps do not need any separate treatment.

Now, by applying $(B1)$ we obtain $d_{\overline{d}}(a+b, b+a) \leq 0$. Moreover, we can compare trees that have not the same structure. For instance, we can transform for free $r = a$ into $r' = a + a$, and then we obtain $d_{\overline{d}}(r', s) \leq \overline{d}(a,b) + \overline{d}(a,c)$, from where we conclude $d_{\overline{d}}(r,s) \leq \overline{d}(a,b) + \overline{d}(a,c)$. Although it could be the case that we could obtain other "lower bounds" of this distance, as we will discuss later in Sect.3 (page 179).

Next, we present another equivalent definition of our bisimulation distance between processes. We consider processes up-to bisimulation, and following the coinductive approach, we will consider a collection of "distance relations" $\{G_m \mid m \in \mathbb{N}\}$, that are those generated by the SOS-rules below:

$$(1)\ \frac{}{p\ G_n\ p} \quad (2)\ \frac{p\ G_n q}{ap\ G_{n+\overline{d}(b,a)}\ bq} \quad (3)\ \frac{p\ G_n\ p'}{p+q\ G_n\ p'+q} \quad (4)\ \frac{p\ G_n\ q \quad q\ G_{n'} r}{p\ G_{n+n'}\ r}$$

Proposition 1. *For all $n \in \mathbb{N}$, we have $p\, G_n\, q$ if and only if $d_{\overline{d}}(p,q) \leq n$.*

Proof. It is clear the correspondence between the rules defining both collections of relations. We will only remark that $(C4)$ corresponds to working up-to bisimilarity, while rule (2) covers both $(C1)$ and $(C3)$ at the same time. □

Remark 1. It would be possible to mix these rules in several ways, even reducing its total number. But we prefer this presentation, where basic transformations are shown in isolation. This definitely simplifies the rule-induction proofs in the following.

3 Simulation Distance

Starting from the bisimulation distance presented above, next we introduce the simulation distance. We start by recalling the definition of simulation.

Definition 4. *A simulation is a relation S between processes such that whenever we have pSq, for every $a \in Act$, if $p \xrightarrow{a} p'$ then, there exists some q', such that $q \xrightarrow{a} q'$ and $p'Sq'$. We say that process p is simulated by process q, or that q simulates p, written $p \sqsubseteq_S q$, if there exists a simulation S such that pSq.*

We want to define by means of rules the relations that indicate how far away is a process p of being simulated by another q. Of course, when q simulates p, the simulation distance between them (in this direction) will be 0. When this is not the case, we will need to change the tree that represents q, to get a process that simulates p, paying for each modification.

Definition 5. *Given two processes p and q, we say that the simulation distance from q to p is at most $m \in \mathbb{N}$, w.r.t. the symmetric distance between actions $\overline{d}$, and then we write $d^S_{\overline{d}}(p,q) \leq m$, if we can derive $p\, G^S_m\, q$ applying the following rules:*

$$(1)\ \frac{p \sqsubseteq_S q}{p\ G^S_n\ q} \quad (2)\ \frac{p\, G^S_n\, q}{ap\ G^S_{n+\overline{d}(b,a)}\ bq} \quad (3)\ \frac{p\, G^S_n\, p'}{p+q\ G^S_n\ p'+q} \quad (4)\ \frac{p\, G^S_n\, q \quad q\, G^S_{n'}\, r}{p\ G^S_{n+n'}\ r}$$

In other words, we can say that the simulation distance is obtained by computing the bisimulation distance up to the similarity relation. This can also be expressed in a transformational way: we look for the "minimal changes" that we need to make in q to get a process q' which simulates p.

Remark 2. Note that in this case we do not need to explicitly say that we work up-to bisimilarity, since when $q \sim q'$, we also have $q' \sqsubseteq_S q$, and then by applying (1) we can transmute q into q' for free, whenever this is needed.

Next we present a very simple example to illustrate how our definition works.

Example 1. We consider the lexicographic distance between actions induced by the lexicographic order, so we have $\overline{d}(a,b) = 1$, $\overline{d}(a,c) = 2$, and so on. Let us consider the processes $p = a(b+c)$ and $q = ab + ad$. Then, it is easy to see that $p \not\sqsubseteq_S q$ and $q \not\sqsubseteq_S p$. Let us start seeing how far away we are of having $q \sqsubseteq_S p$. It is clear that $q \sqsubseteq_S p'$, where p' is obtained from p by turning c into d, so that we define $p' = a(b+d)$. Therefore, we have $d^S_{\overline{d}}(p,q) \leq \overline{d}(c,d) = 1$. Next we see in detail how we can derive $q\, G_1^S\, p$ applying the rules in Def.5:

$$\dfrac{\dfrac{q \;\sqsubseteq_S\; p'}{q\;\; G_0^S\;\; p'}(1) \qquad \dfrac{\dfrac{d\;\; G^S_{\overline{d}(c,d)}\;\; c}{b+d\;\; G_1^S\;\; b+c}(3)}{p'\;\; G^S_{1+\overline{d}(a,a)}\;\; p}(2)}{q\;\; G_1^S\;\; p}(4)$$

If we consider the opposite distance, which measures at which extent we have (not) $p \sqsubseteq_S q$, the shortest way to obtain some q' with $p \sqsubseteq_S q'$ is to duplicate (for free) the subtree below a, and then we change one of the b actions into c, paying for it $\overline{d}(b,c)$. So we obtain $q' = a(b+c)+ad$, which produces $d_{\overline{d}}(q,p) \leq \overline{d}(b,c) = 1$. This can be inferred applying our rules as follows:

$$\dfrac{\dfrac{p \;\sqsubseteq_S\; q'}{p\;\; G_0^S\;\; q'}(1) \qquad \dfrac{\dfrac{\dfrac{c\;\; G_1^S\;\; b}{b+c\;\; G_1^S\;\; b+b}(3)}{b+c\;\; G_1^S\;\; b}(B2)}{a(b+c)\;\; G^S_{1+\overline{d}(a,a)}\;\; ab}(2)}{p\;\; G_1^S\;\; q}(4)$$

Next we compare the definitions of simulation distance based on the simulation game with ours.

Definition 6. *(Simulation game) Given two LTSs, L_1 and L_2, we call configurations the pairs (p,q), with $p \in L_1$ and $q \in L_2$. The simulation game is played by two players: the attacker $\mathbb{A}$ and the defender $\mathbb{D}$. The initial configuration of the game deciding if $p_0 \sqsubseteq_s q_0$, is just the pair (p_0, q_0). A round of the game, when the current configuration is (p,q), proceeds as follows:*

1. *$\mathbb{A}$ chooses a transition in L_1: $p \xrightarrow{a} p'$.*
2. *$\mathbb{D}$ must execute the same action at the other side of the board (L_2): $q \xrightarrow{a} q'$.*
3. *The game proceeds in the same way from the new configuration (p', q').*

The winner of the game is defined by the following rules: (1) Any infinite game is a win for $\mathbb{D}$. (2) $\mathbb{D}$ also wins if $\mathbb{A}$ cannot make any new move. (3) $\mathbb{A}$ wins when he makes a move, that $\mathbb{D}$ cannot reply with a transition from L_2.

Theorem 1. *$p \sqsubseteq_S q$ (resp. $p \not\sqsubseteq_S q$) if and only if $\mathbb{D}$ (resp. $\mathbb{A}$) has a winning strategy for the simulation game starting at (p,q).*

The simulation game can be turned into a (classical) simulation distance game by allowing to reply any a-move by some b-move with $b \neq a$, but then the defender

should pay $\overline{d}(b,a)$ to the attacker for the mismatch. The value of the game provides the "classical" simulation distance between p and q [1]. We can obtain a coinductive characterization, which also provides a more general definition covering also infinite processes, as follows:

Definition 7. *A family of relations between processes $(S_n)_{n\in\mathbb{N}}$ is a classical simulation distance family (csdf), w.r.t. the symmetric distance between actions $\overline{d}$, when for each $(p,q) \in S_n$ we have the diagram:*

$$\begin{array}{ccc} p & S_n & q \\ \forall\ a\downarrow & \Longrightarrow & \downarrow\exists b \\ p' & S_{n-\overline{d}(b,a)} & q' \end{array}$$

We say that p and q are at most at classical simulation distance n, and then we write $d^S_{\overline{d}}(p,q) \leq n$, iff there is some csdf $(S_n)_{n\in\mathbb{N}}$ *such that pS_nq.*

Example 2. Using the distance relation $\overline{d}$ at Example 1, if we apply our Def.5, we get $d^S_{\overline{d}}(a+d, b+e) \leq 2$, but we cannot obtain $d^S_{\overline{d}}(a+d, b+e) \leq 1$. Instead, we can get a *csdf* taking $S_1 = \{(a+d, b+e)\}$ and $S_0 = \{(\mathbf{0},\mathbf{0})\}$, because $a+d \xrightarrow{a} \mathbf{0}$ can be replied by $b+e \xrightarrow{b} \mathbf{0}$ with cost 1. If we consider the discrete distance $\overline{d}$ defined by $\overline{d}(a,b) = 1 \Leftrightarrow a \neq b$, then we obtain $d_{\overline{d}}(\sum_{i=1}^n a_i, a_0) \leq n$, but $d_{\overline{d}}(\sum_{i=1}^n a_i, a_0) \not\leq n-1$, while using the classical simulation game approach we can take $S_1 = \{(\sum_{i=1}^n a_i, a_0) \mid n \in \mathbb{N}\}$ and $S_0 = \{\mathbf{0},\mathbf{0}\}$, because any move $\sum a_i \xrightarrow{a_i} \mathbf{0}$ can be replied by $a_0 \xrightarrow{a_0} \mathbf{0}$ with cost 1.

Even if we consider that our "global simulation distance", defined at Def.5, is the most adequate way to turn the simulation relation into a quantitative distance between processes, next we will show the flexibility of our approach showing that a simple variation of the system of rules defining it produces a characterization of the "classical" operational simulation distance, defined at Def.7. We only need to change rule (3), taking instead the new rule (3′), thus obtaining the revised system:

$$(1)\ \frac{p \sqsubseteq_S q}{p\ H^S_n\ q}\quad (2)\ \frac{p\,H^S_n q}{ap\ H^S_{n+\overline{d}(b,a)}\ bq}\quad (3')\ \frac{p\,H^S_n\,p'\ \ q\,H^S_{n'}\,q'}{p+q\ H^S_{max\{n,n'\}}\ p'+q'}\quad (4)\ \frac{p\,H^S_n\,q\ \ q\,H^S_{n'}r}{p\ H^S_{n+n'}\ r}$$

We will see that the use of max in this rule produces that only the cost of the simulation of the computation that is "harder to simulate" is taken into account when generating the relations H^S_n. As a consequence, the family $(H^S_n)_{n\in\mathbb{N}}$ is a *csdf* that accurately generates the classical simulation distance:

Theorem 2. *1. $(H^S_n)_{n\in\mathbb{N}}$ is a csdf.*
2. If $(S_n)_{n\in\mathbb{N}}$ is a csdf then $S_n \subseteq H^S_n$.

Proof. • 1| We prove that $(H^S_n)_{n\in\mathbb{N}}$ satisfies the definition of *csdf*, by rule induction on the definition of H^S_n:

$$\begin{array}{lcccl} (1): & p & H_n^S & q & \\ & \forall a\downarrow & & \downarrow\exists\ b{=}a & (\overset{df}{\Leftarrow} p \sqsubseteq_S q) \\ & p' & H_n^S & q' & (\overset{(1)}{\Leftarrow} p' \sqsubseteq_S q') \end{array}$$

$$\begin{array}{lcccl} (2): & ap & H^S_{n+\overline{d}(b,a)} & bq & \\ & a\downarrow & & \downarrow b & \\ & p & H^S_{n+\overline{d}(b,a)-\overline{d}(b,a)} & q & \\ & & \Downarrow & & \\ & p & H_n^S & q & \end{array} \qquad \begin{array}{lcccl} (3'): & p+q & H_n^S & p'+q' & \\ & a\downarrow & & \downarrow b & \\ & p'' & H^S_{n-\overline{d}(b,a)} & p''' & \\ & & \Uparrow pH_n^S p' \wedge qH^S_{n'}q' \ with\ n{\geq}n' & & \\ & p & H_n^S & p' & \\ & a\downarrow & & \downarrow b \quad (by \quad i.h.) & \\ & p'' & H^S_{n-\overline{d}(b,a)} & p''' & \end{array}$$

$$\begin{array}{lcccl} (4): & p & H^S_{n+n'} & r & \\ & a\downarrow & & \downarrow c & (\Leftarrow i.h.(4)) \\ & p' & H^S_{n+n'-(\overline{d}(c,b)+\overline{d}(b,a))} & r' & \\ & & \Downarrow \overline{d}(c,a)\leq\overline{d}(c,b)+\overline{d}(b,c) & & \\ & p' & H^S_{n+n'-\overline{d}(c,a)} & r' & \end{array} \qquad \begin{array}{lcccl} p & H_n^S & q & q & H^S_{n'} \quad r \\ a\downarrow & & \downarrow b & b\downarrow & \downarrow c \\ p' & H^S_{n-\overline{d}(b,a)} & q' & q' & H^S_{n'-\overline{d}(c,b)} \quad r' \end{array}$$

- 2| We use complete induction on the depth of p:

$$0\ S_n\ q\ \Rightarrow\ 0\ \sqsubseteq_s\ q\ \Rightarrow\ 0\ H_n^S\ q$$

Let $p = ap'_a + r$ and $q = bq'_b + q''$ such that

$$\begin{array}{ccccc} p = ap'_a + r & & S_n & & q = bq'_b + q'' \\ \forall\ a\downarrow & & \Longrightarrow & & \downarrow\exists\ b \\ p_a & & S_{n-d(b,a)} & & q'_b \end{array}$$

Then we have:

$$p'_a\ S_{n-d(b,a)}\ q'_b\ \Rightarrow\ p'_a\ H^S_{n-d(b,a)}\ q'_b\ \Rightarrow\ ap'_a\ H_n^S\ bq'_b\,.$$

This happens for all the summands of p, which means that up-to idempotence of $+$, we can assume that $p = \sum a_i p'_{a_i}$ and $q = \sum b_i q'_{b_i} + r$, where for all $i \in I$ we have $a_i p'_{a_i}\ H_n^S\ b_i q'_{b_i}$; and finally we conclude $p\ H_n^S\ q$, by applying repeatedly the rule (3), and (1) to get $\mathbf{0}\, H_n^S\, r$. □

It is interesting to note that we have not used the transitivity rule (4) at all in the previous proof, which means that we can obtain the following corollary:

Corollary 1. *If we define $H_n^{S'}$ as H_n^S, but removing the transitivity rule (4), we have that $H_n^{S'}$ is equivalent to H_n^S.*

Proof. From the fact that H_n^S is a *csdf* we immediately obtain that $H_n^{S'}$ is too. But since in the proof of Th.2 we do not use the transitivity rule (4), we have also proved there that for any *csdf* $(S_n)_{n\in\mathbb{N}}$ we have $S_n \subseteq H_n^{S'}$. Then we have $H_n^S \subseteq H_n^{S'}$ and from their definitions we immediately obtain $H_n^{S'} \subseteq H_n^S$, from where we can conclude that H_n^S is equivalent to $H_n^{S'}$. □

Note however, that when we consider the sum between branches in rule (3) instead of the maximum, as done in Def.5, we need indeed the transitivity rule, because in this case it cannot be "derived" from the rest of the rules. The following example shows the necessity of this rule.

Example 3. Consider the processes $p = a$ and $q = b + c$, if we want to simulate q by p, we need to change action a into both b and c. However, it is possible that it would be better to transform first a into some a', and then this a' into b and c. Without the transitivity rule we cannot generate this elaborated transformation, and then we would not get the "desired" global simulation distance. Instead, when we consider the classical simulation distance, by the triangular inequality, it is not useful to transform first a into some a' and then a' into b, because that will be always worse than transforming directly a into b.

This example also illustrates the possible interest of such an elaborated procedure in order to efficiently simulate several branches of the simulated process by a common branch of the simulating one. The cost of the transformation of a into a' is shared by the two branches, and then we only pay once for it. Note that the use (for free) of idempotence allows this double use of a common branch.

4 Bisimulation Distance

Using the bisimulation game, we can define a "classical" bisimulation distance as done in [7]. It measures how far away are two processes of being bisimilar.

Theorem 3 ([10, 12]). *$p \sim q$ (resp. $p \not\sim q$) if and only if $\mathbb{D}$ (resp. $\mathbb{A}$) has a winning strategy for the bisimulation game starting at (p, q).*

Definition 8. *A family $(R_n)_{n\in\mathbb{N}}$ is a classical bisimulation distance family (cbdf), w.r.t. the symmetric distance relation between actions $\overline{d}$, when it satisfies*

$$\begin{array}{ccc} p & R_n & q \\ \forall\, a\downarrow & \Longrightarrow & \downarrow\exists b \\ p' & R_{n-\overline{d}(b,a)} & q' \end{array} \qquad \wedge \qquad \begin{array}{ccc} p & R_n & q \\ \exists a\downarrow & \Longleftarrow & \downarrow\forall\, b \\ p' & R_{n-\overline{d}(a,b)} & q' \end{array}$$

We say that p and q are at most at classical bisimulation distance n, and then we write $d_{\overline{d}}^B(p, q) \leq n$, iff there is some cbdf $(R_n)_{n\in\mathbb{N}}$ *such that pR_nq.*

From the symmetric definition of bisimulation we immediately obtain that our classical bisimulation distance is also symmetric.

Proposition 2. *For any two processes p, q and any $n \in \mathbb{N}$, we have $d_{\overline{d}}^B(p,q) \leq n$ if and only if $d_{\overline{d}}^B(q,p) \leq n$.*

Following the same ideas that we used in Sect.3, we can obtain a rule system that produces the biggest relations H_n^B that state that the related processes are at most at distance n to be bisimilar.

Definition 9. *We consider the family of relations $(H_n^B)_{n\in\mathbb{N}}$ which are generated by applying the following rules, modulo bisimulation:*

$$(1)\ \frac{}{p\ H_n^B\ p} \quad (2)\ \frac{p\ H_n^B\ q}{ap\ H_{n+\overline{d}(b,a)}^B\ bq} \quad (3')\ \frac{p\ H_n^B\ p' \quad q\ H_{n'}^B\ q'}{p+q\ H_{max\{n,n'\}}^B\ p'+q'} \quad (4)\ \frac{p\ H_n^B\ q \quad q\ H_{n'}^B\ r}{p\ H_{n+n'}^B\ r}$$

It is nice to observe the close similarity between the rules defining this classical bisimulation distance and our previous bisimulation distance in Sect.2: in fact, if we change the max operator in $(3')$ by addition, then it is easy to check that the obtained definition is equivalent to our original one.

Remark 3. It is clear that we can remove the "up-to" bisimulation at the definition above if we explicitly introduce the bisimilarity relation in the definition, by replacing rule (1) by the following rule:

$$(1')\ \frac{p \sim q}{p\ H_n^B\ q}$$

However, we prefer our first presentation in order to stress the fact that the system of rules that defines the classical simulation distance is obtained from the one above simply adding the similarity relation to produce pairs that are "0-far" away.

We can prove the relationship between the family H_n^B defined above and the "classical" bisimulation distance relations defined at Def.8, exactly as we made for the simulation case.

Theorem 4. *1. $(H_n^B)_{n\in\mathbb{N}}$ is a cbdf.*
2. If $(R_n)_{n\in\mathbb{N}}$ is a cbdf then $R_n \subseteq H_n^B$.

Once again, we do not use rule (4) at the proof above, which allows to derive the following corollary, that is analogous to Cor.1 in Sect.3.

Corollary 2. *If we define $H_n^{B'}$ as H_n^B in Def.9, but removing the transitivity rule (4), then we obtain the same family of relations, that is $H_n^{B'} = H_n^B$, $\forall n \in \mathbb{N}$.*

5 Distances for All the Semantics in the ltbt-Spectrum

Inspired by the connection between the bisimulation and the simulation distances, next we define a general notion of distance between processes. It can be instantiated by any of the different semantics in the ltbt-spectrum. These distances will measure how far away is any process q of being greater than p

with respect to each of the semantic preorders defining the semantics in Fig.1. Roughly speaking, to obtain these distances, we compute the cost of changing some actions in both p and q in order to obtain two new processes p' and q' which are related under the considered semantics.

We could try to base our general definitions on the "classical" simulation distance. It is defined in a similar way as the "classical" bisimulation distance. The only difference between those two definitions was the use of $\sqsubseteq_S$ at rule (1). This immediately suggests us to define the semantic distances, corresponding to any semantics defined by an order $\sqsubseteq_{\mathcal{L}}$, by means of the following system of rules:

$$(1)\ \frac{p \sqsubseteq_{\mathcal{L}} q}{p\ H^{\mathcal{L}}_n\ q}\quad (2)\ \frac{p\ H^{\mathcal{L}}_n q}{ap\ H^{\mathcal{L}}_{n+\overline{d}(b,a)}\ bq}\quad (3)\ \frac{p\ H^{\mathcal{L}}_n\ p'\ \ q\ H^{\mathcal{L}}_{n'}\ q'}{p+q\ H^{\mathcal{L}}_{max\{n,n'\}}\ p'+q'}\quad (4)\ \frac{p\ H^{\mathcal{L}}_n\ q\ \ q\ H^{\mathcal{L}}_{n'}\ r}{p\ H^{\mathcal{L}}_{n+n'}\ r}$$

However, when checking some simple examples we see that this "local" approach (based on max) does not produce a "reasonable" distance for some of the most popular semantics in the ltbt-spectrum. Next, we consider the case of ready simulation (RS).

Example 4. Let us consider the processes $p = b + c$ and $q = d + f$. As distance relation $\overline{d}$ between actions, we consider again the lexicographic distance. We can check that the definition above produces

$$\frac{\dfrac{b\ H^{RS}_2\ d \quad c\ H^{RS}_3\ f}{b+c\ H^{RS}_{max\{2,3\}}\ d+f}(3)}{p\ \ H^{RS}_3\ \ q}(df)$$

We infer $p\ H^{RS}_3\ q$, that is the result of the necessary change in the branch which needs the most expensive change. However, this is, by no means, consistent with the definition of ready simulation: In order to have $p \sqsubseteq_{RS} q$, we need that the two processes have the same initial offer. Therefore, we would need to transform the offer $\{d, f\}$ into $\{b, c\}$. We would need changes whose aggregated cost would be (at least) 4—see Example 5—, and not just 3.

Note that this problem does not appear in the simulation case, because the definition of simulation does not contain any "global" factor. But, most of the rest of the semantics, take somehow into account some "global" information that could only be obtained by combining the information taken from several separated computations. This is the case of ready sets at readiness semantics, or even the case of failures defining the failure semantics.

Certainly, we also had $p\ H^{B}_3\ q$ for the (classical) bisimulation distance, and then we should also expect $p\ H^{\mathcal{L}}_3\ q$ for any semantics coarser than bisimulation. But as we discussed at the end of our introduction, plain bisimilarity is able to check the equality of the offers of two processes even if working in a local way. However, once we need to compare two unequal offers, this local procedure proves to be quite limited. Therefore, we need to recover our first proposal at Sect.2 that measures the distance between processes by adding the cost of all the changes that we have to do at all the branches of the tree that represents a process. We already saw that it provides two reasonable "global" notions of simulation and

bisimulation distances. Based on it, we obtain our general definition of "global" semantic distance between processes:

Definition 10. *Given a semantics $\mathcal{L}$, defined by a preorder $\sqsubseteq_{\mathcal{L}}$, we say that a process q is at global distance at most $m \in \mathbb{N}$ of being better than some other p, w.r.t. the semantics $\mathcal{L}$ and the distance between actions $\overline{d}$, and then we write $gd^{\mathcal{L}}_{\overline{d}}(p,q) \leq n$, if we can infer $p\ G^{\mathcal{L}}_n\ q$, by applying the following rules:*

$$(1)\ \frac{p \sqsubseteq_{\mathcal{L}} q}{p\ G^{\mathcal{L}}_n\ q} \quad (2)\ \frac{p\ G^{\mathcal{L}}_n\ q}{ap\ G^{\mathcal{L}}_{n+\overline{d}(b,a)}\ bq} \quad (3)\ \frac{p\ G^{\mathcal{L}}_n\ p'}{p+q\ G^{\mathcal{L}}_n\ p'+q} \quad (4)\ \frac{p\ G^{\mathcal{L}}_n\ q \quad q\ G^{\mathcal{L}}_{n'}\ r}{p\ G^{\mathcal{L}}_{n+n'}\ r}$$

Example 5. It is easy to check that for the processes in Example 4 and the ready simulation semantics RS, we obtain now the desired distance $gd^{RS}_{\overline{d}}(p,q) \leq 4$, since we can infer applying the rules for $\mathcal{L} = RS$ that:

$$\dfrac{\dfrac{\dfrac{b\ G^{RS}_1\ c}{b+c\ G^{RS}_1\ c+c}(3) \quad \dfrac{\dfrac{\dfrac{c+c \sqsubseteq_{RS} c}{c+c\ G^{RS}_0\ c}(1) \quad \dfrac{c\ G^{RS}_1\ d \quad \dfrac{d \sqsubseteq_{RS} d+d}{d\ G^{RS}_0\ d+d}(1)}{c\ G^{RS}_{1+0}\ d+d}(4)}{c+c\ G^{RS}_{0+1}\ d+d}(4)}{b+c\ G^{RS}_{1+1}\ d+d}(4) \quad \dfrac{d\ G^{RS}_2\ f}{d+d\ G^{RS}_2\ d+f}(3)}{b+c\ G^{RS}_{2+2}\ d+f}(4)}{p\ G^{RS}_4\ q}(df)$$

Remark 4. As a matter of fact, we have only used rule (1) in the partial case of "idempotence". This means that the computed (bound of the) distance will also be valid for the bisimulation semantics and in fact for any other semantics in the spectrum. Of course, if we consider a coarser semantics, it could be the case that we could obtain a smaller distance by applying (1) in some other way. For instance, for the simulation semantics (S) we will easily obtain $gd^{S}_{\overline{d}(p,q)} \leq 2$.

Generally, we immediately obtain the following result that asserts that our family of distances reflects exactly the hierarchy in the ltbt-spectrum.

Proposition 3. *Whenever we have two semantics $\mathcal{L}_1$ and $\mathcal{L}_2$ and the first is finer than the latter ($\sqsubseteq_{\mathcal{L}_1} \subseteq \sqsubseteq_{\mathcal{L}_2}$), then we have $gd^{\mathcal{L}_1}_{\overline{d}}(p,q) \leq n \Rightarrow gd^{\mathcal{L}_2}_{\overline{d}}(p,q) \leq n$, for all processes p, q and any value $n \in \mathbb{N}$.*

6 Generalizations, Applications and Some Conclusions

In the developments above we have preferred to consider symmetric distances between actions because in particular we wanted to apply all the notions and technical definitions to the case of bisimulation, that is an equivalence relation and therefore symmetric. However, the rest of the semantics are typically defined by means of a preorder, instead of by an equivalence relation. This is why the consideration of asymmetric quasi-distances opens a new and quite interesting space for developments and applications of our theory.

Let us consider the case of the simulation semantics: when we have $p \sqsubseteq_S q$, this reflects that q has all the capabilities of p and possibly some others. The

simulation distances presented above reflect how many changes we need to make in q in order to get a process that really simulates p. But it could be the case that q instead of directly offering the same actions offered by p, offers some others that we consider that "do perfectly the work". This situation is formally covered simply by replacing the symmetric distance between actions by an asymmetric quasi-distance, defined as follows:

Definition 11. *An asymmetric quasi-distance in a set of actions Act is a function* $d : Act \times Act \to \mathbb{N}$ *which satisfies* $d(a, a) = 0\ \forall a \in Act$*, and the triangular inequality* $d(a, b) + d(b, c) \geq d(a, c)\ \forall a, b, c \in Act$*. We will say that* $d(a, b)$ *expresses "how far away" is action* a *of covering the expectation to have a* b*.*

Remark 5. Now we can have $d(b, a) = 0$ even if $b \neq a$, and this would reflect the fact that b totally "simulates" a. Then we could replace without "cost" any occurrence of an action a in the simulated process p using the action b. Of course, now we can have $d(a, b) \neq d(b, a)$, because the cost of replacing a by b could be very different from that of replacing b by a. Finally, any asymmetric quasi-distance induces a symmetric quasi-distance, simply taking $\overline{d}(a, b) = max\{d(a, b), d(b, a)\}$. This becomes a distance if we impose that $a \neq b \Rightarrow \overline{d}(a, b) \neq 0$.

Example 6. If we consider a simple vending machine that returns no change, and a product costs 1€, then from the machine point of view a payment of 2€ for it, could be perfectly assumed. Instead, if the situation is the other way around and we pay 1€ for a product whose cost is 2€, then the company loses 1€. This would be reflected by the asymmetric quasi-distance defined by d(1€,2€) = 0 and d(2€,1€) = 1. Using it we obtain that the process where we pay 2€ instead of 1€ is at distance 0 of simulating the specification, while when we pay 1€ when a 2€ cost is specified, we would be at distance 1 of satisfying the specification.

Using the fact that all the semantics in the (extended) ltbt-spectrum are connected to some constrained simulation, we could justify the consideration of the corresponding "biased" distances. Instead, it seems not possible to define a reasonable bisimulation distance really based on an asymmetric quasi-distance. Of course, we could always do the task using the induced distance $\overline{d}$, but in this way we are "loosing" the asymmetric information in the original distance d.

We have defined our distances with natural values just to simplify the presentation, but there is no problem at all on using any other totally ordered set, such as $\mathbb{R}^+$. Moreover, if we use fixed values for the weight of any discordance along a computation (or at any place of the trees when considering "global" distances) then the distance between two (infinite) processes would become infinite as soon as the number of discordances between them is also infinite. This would be certainly a problem, for instance, when comparing cyclic programs where any discordance will appear again at any iteration of the compared processes. Of course, the solution to this problem would consist (as proposed, e.g. in [4, 13]) on defining weighted distances. For them the weight of any disagreement at the n-th step of a computation (or at the n-th level of the unfolded processes) will decrease fast enough (for instance, the classical weights used at the literature are those defined by the exponential sequence $\frac{1}{2^n}$).

It is true, however, that we have not discussed how to obtain in a precise way the (bounds for the) distances between two infinite processes, when they "disagree" at infinitely many places. This could be done by using either finite approximations or recursion-induction rules, for the case of finite state processes. But certainly the details need a careful work.

A simpler extension solves the problem of unexpected termination. If we consider for instance our Def.3, we could extend it by adding a fixed payment f, for unexpected termination, taking $d(p, \mathbf{0}) \leq f$ and $d(\mathbf{0}, p) \leq f$, $\forall p \neq \mathbf{0}$. Instead, we could pay for each of the lost actions a quantity q_a, taking $\overline{d}(a\mathbf{0}, \mathbf{0}) \leq q_a$ and $\overline{d}(\mathbf{0}, a\mathbf{0}) \leq q_a$ $\forall a \in Act$. Of course, this second possibility would produce infinite distances if the terminated process was infinite, but weights can be also introduced here if we want to follow this approach.

We consider that starting from the basic (but quite flexible) definitions introduced in this paper we are plenty of more elaborated possibilities, which could be developed by adapting the ideas in our general theory to them. Next, we give a list of interesting directions that we expect to explore in the near future. First, we are working in a definition of *approximated testing*, where we indicate "at which extent" a process passes a test. Using this notion we can quantify the testing procedure by formalizing the quite frequent situation in practice where the specification states the *ideal* behavior of the desired implementations, but some small disagreements are tolerated by the *quality* standards. A dual application of our distances would also provide for free a nice quantification of the notion of *robustness*: given some specification p we would say that a given implementation q is n-robust w.r.t. some semantics $\mathcal{L}$ when any "n-wrong" behavior of q, that is, any q' such that $d_x^{\mathcal{L}}(q', q) \leq n$, satisfies $d_x^{\mathcal{L}}(p, q') = 0$. We can combine our approximated correctness and the quantified robustness proposed above, to define a notion of approximated robustness, where we also allow some small disagreement between p and the n-wrong behaviors of q.

Another generalization would use "contextually defined" distances between actions, that take into account the fact that several occurrences of the same action in a specification could play totally different roles. In such a case, we could specify at each state of the specification which is the distance between actions that we should use locally at each place. The distances between *pure trees*, where the application of the idempotence law is not allowed, will also capture *redundancy*, and then when investigating fault tolerance the previously discussed ideas on approximated robustness could be used to define *approximated fault tolerance*.

Finally, we could also allow negative values at the distances between actions, that would state that whenever we have $d(a, b) = -n$ then using b to simulate a we would be "improving" the quality of the system. This could amortize some other steps where we have the opposite situation. A typical application would appear when comparing two transmission protocols, and is clearly related with the previous work by Vogler and Lüttgen in [9], where "faster than" preorders where studied, and those by Kiehn and Arun-Kumar [8] on *amortized bisimulation*.

References

[1] Černý, P., Henzinger, T.A., Radhakrishna, A.: Quantitative Simulation Games. In: Manna, Z., Peled, D. (eds.) Time for Verification. LNCS, vol. 6200, pp. 42–60. Springer, Heidelberg (2010)

[2] Černý, P., Henzinger, T.A., Radhakrishna, A.: Simulation Distances. In: Gastin, P., Laroussinie, F. (eds.) CONCUR 2010. LNCS, vol. 6269, pp. 253–268. Springer, Heidelberg (2010)

[3] Chen, X., Deng, Y.: Game Characterizations of Process Equivalences. In: Ramalingam, G. (ed.) APLAS 2008. LNCS, vol. 5356, pp. 107–121. Springer, Heidelberg (2008)

[4] de Alfaro, L., Faella, M., Stoelinga, M.: Linear and branching system metrics. IEEE Trans. Software Eng. 35(2), 258–273 (2009)

[5] de Frutos-Escrig, D., Gregorio-Rodríguez, C., Palomino, M.: On the unification of process semantics: equational semantics. ENTCS 249, 243–267 (2009)

[6] de Frutos Escrig, D., Gregorio Rodríguez, C., Palomino, M.: On the Unification of Process Semantics: Observational Semantics. In: Nielsen, M., Kučera, A., Miltersen, P.B., Palamidessi, C., Tůma, P., Valencia, F. (eds.) SOFSEM 2009. LNCS, vol. 5404, pp. 279–290. Springer, Heidelberg (2009)

[7] Fahrenberg, U., Legay, A., Thrane, C.R.: The quantitative linear-time–branching-time spectrum. In: Chakraborty, S., Kumar, A. (eds.) FSTTCS. LIPIcs, vol. 13, pp. 103–114. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2011)

[8] Kiehn, A., Arun-Kumar, S.: Amortised Bisimulations. In: Wang, F. (ed.) FORTE 2005. LNCS, vol. 3731, pp. 320–334. Springer, Heidelberg (2005)

[9] Lüttgen, G., Vogler, W.: Safe Reasoning with Logic LTS. In: Nielsen, M., Kučera, A., Miltersen, P.B., Palamidessi, C., Tůma, P., Valencia, F. (eds.) SOFSEM 2009. LNCS, vol. 5404, pp. 376–387. Springer, Heidelberg (2009)

[10] Nielsen, M., Clausen, C.: Bisimulation, Games, and Logic. In: Karhumäki, J., Rozenberg, G., Maurer, H.A. (eds.) Results and Trends in Theoretical Computer Science. LNCS, vol. 812, pp. 289–306. Springer, Heidelberg (1994)

[11] Stirling, C.: Modal and Temporal Logics for Processes. In: Moller, F., Birtwistle, G. (eds.) Logics for Concurrency. LNCS, vol. 1043, pp. 149–237. Springer, Heidelberg (1996)

[12] Stirling, C.: Bisimulation, modal logic and model checking games. Logic Journal of the IGPL 7(1), 103–124 (1999)

[13] Thrane, C.R., Fahrenberg, U., Larsen, K.G.: Quantitative analysis of weighted transition systems. J. Log. Algebr. Program. 79(7), 689–703 (2010)

[14] van Glabbeek, R.: The linear time-branching time spectrum I: the semantics of concrete, sequential processes. In: Bergstra, J.A., Ponse, A., Smolka, S.A. (eds.) Handbook of Process Algebra, ch. 1, pp. 3–99. Elsevier (2001)

[15] Winskel, G.: Synchronisation Trees. In: Díaz, J. (ed.) ICALP 1983. LNCS, vol. 154, pp. 695–711. Springer, Heidelberg (1983)

6.2 An algebraic approach

In 2012 we also presented the following paper at the international workshop *Recent Trends in Algebraic Development Techniques (WADT)*, held in Salamanca (Spain). Our idea was to define our distances between processes in a general algebraic framework. We started by computing the distance between processes, and afterwards we saw them as a (finite) term for a given (general) signature. The approach is applied over BCCSP showing again that it can cover all the semantics in the ltbt-spectrum. Our proposals also covered the classical bisimulation distance, so that the obtained framework was in accordance with our work presented in Section 6.1 as well as those by other authors.

Our starting point was the algebraic semantics developed by M. Hennessy in his book *Algebraic Theory of Processes*. The objective was to obtain an alternative definition of our distances which allowed the use of classical algebraic tools to study them. We also included an algebraic characterization of the bisimulation game with distances, illustrating the generality of our approach.

Distances between Processes: A Pure Algebraic Approach*

David Romero Hernández and David de Frutos Escrig

Dpto. Sistemas Informáticos y Computación
Facultad CC. Matemáticas, Universidad Complutense de Madrid, Spain
dromeroh@pdi.ucm.es, defrutos@sip.ucm.es

Abstract. Recently, we have presented operational and denotational definitions for distances between processes corresponding to any semantics in the ltbt-spectrum. In this paper, we develop a general algebraic framework to define distances between terms from any arbitrary signature. We apply this framework obtaining a new algebraic characterization of our previous distances. Moreover, we prove the generality of our approach developing an algebraic characterization of the distances based on the (bi)simulation game by other authors.

1 Introduction

In order to define an (abstract) semantics for processes, we need just to define an adequate equivalence relation, $\equiv$, relating the processes in some universe, *Proc*. Then, the values of this semantics are just the corresponding equivalence classes, and two processes have the same semantics if and only if they are equivalent. Once we have fixed such a semantics we can compare two processes, but the output of this comparison is just a Boolean value. In particular, when two processes are not equivalent, we do not have a general way to measure "how far away" they are from being equivalent.

There are several papers which introduce several distances between processes based on the (bi)simulation game –see e.g. [3, 5]–. Even before, Ying and Wirsing studied approximate bisimilarity, following similar but simpler ideas [12, 11]. Our work started by considering those distance games, where one essentially plays the (bi)simulation game, but with the "defender" having the possibility of replying a move of the "attacker" without matching exactly his move. In such a case, he should pay to the attacker some quantity, depending on the mismatch (distance) between the two involved actions.

It is well-known the use of equivalence relations to formalize the notion of implementation: a process implements some specification (given by another process) when they are equivalent w.r.t. the adequate semantics. But, if we follow this approach, we have no flexibility at all: our process has to satisfy in a precise

* Partially supported by the Spanish projects TESIS (TIN2009-14312-C02-01), DESAFIOS10 (TIN2009-14599-C03-01) and PROMETIDOS S2009/TIC-1465.

N. Martí-Oliet and M. Palomino (Eds.): WADT 2012, LNCS 7841, pp. 265–282, 2013.

way all the constraints imposed by the specification, or it will not be a correct implementation. Instead, in the real world, we often find other more flexible quality requirements, where the specification establishes the ideal behavior of the system, but some (limited) deviations from it are allowed, without invalidating the adequateness of the implementation.

We need a notion of distance between processes to make precise how far away two processes are from being equivalent w.r.t. some given semantics. It is true that metrics have been used for a long time to formalize the semantics of infinite processes, by means of (the limit of) those of their finite approximations. But these metrics were just a very particular case that only cared for "the first" disagreement between the compared processes. Instead, now we look for more general distances, which moreover should be applicable to any syntactic process algebra (i.e., to any signature Σ) and any semantics (based, for instance, on the axiomatization of the desired semantics).

We have already introduced the basic operational ideas of our approach in [8]. It is true that the most flexible way to capture a semantics for processes, $\mathcal{L}$, is based on the use of an adequate preorder $\subseteq_{\mathcal{L}}$, and not just an equivalence relation $\equiv_{\mathcal{L}}$.

However, we prefer to start our presentation in Section 2 using just the better known *equational calculus*. Next, in Section 3 we will see that a simple modification of the proof system defining the classical equational calculus (see e.g. [7]), produces a general algebraic framework to define distances between processes w.r.t. any semantics. In particular, in Section 4 we study in detail the case of the bisimulation semantics. Later, in Section 5, we will see how we can easily generalize all our algebraic presentation to the inequational framework. Moreover, we define our distances for the rest of the semantics in the ltbt-spectrum. To show the flexibility of our approach, in Section 6 we see how the classic distances based on the (bi)simulation game, can be also defined in our algebraic framework. We conclude with our conclusions and some future work.

2 Preliminaries

A careful presentation of the equational calculus with application to the (testing) semantics of processes can be found in [7]. Next, we will only remind the definitions of the main concepts needed to develop that theory.

Definition 1. *1. A* signature*, Σ, is a set of formal* functional symbols*. Each functional symbol has associated a natural number which is its* arity*. We use Σ_n to denote the set of symbols in Σ of arity n.*

2. If Σ is a signature, a Σ-algebra is a pair $\langle A, \Sigma_A \rangle$ where A is a set, the domain or support of the algebra, and Σ_A is a set of internal functions. A Σ-algebra is simply an interpretation of each operation in Σ, respecting of course its arity.

3. There is a particularly important Σ-algebra, called the term algebra *for Σ, and denoted by T_Σ, whose support is the set of terms freely generated by Σ.*

Some particular collections of Σ-algebras can be singled out by means of equations. An equation is determined by a pair of terms which may contain variables. We consider an (arbitrary) set of variables, X, and the set of terms with variables $T_\Sigma(X)$. $T_\Sigma(X)$ can be obtained by extending the signature Σ adding these variables with null arity. In fact $T_\Sigma(X)$ is just an algebra, where $\Sigma(X)$ is the classic notation for the signature $\Sigma \cup X$ which add to Σ each $x \in X$ as a new function symbol of arity 0.

Given an equation $t \equiv t'$ with $t, t' \in T_\Sigma(X)$, we say that a Σ-algebra $\langle A, \Sigma_A \rangle$ satisfies it, when the values of both t and t' under any valuation, which assigns values in A to the variables $x \in X$, are the same. Given a set of equations E, $\mathcal{C}(E)$, is the class of Σ-algebras satisfying the equations E. The initial algebra of $\mathcal{C}(E)$ can be presented as a quotient algebra $T_\Sigma / \equiv_E$ for some particular congruence $\equiv_E$. We can obtain this congruence by means of the equational deduction system $\mathbf{DED}(E)$ in Fig. 1 whereby the equations in E may be used to derive statements of the form $t \equiv t'$, with $t, t' \in T_\Sigma$.

1. Reflexivity $\dfrac{}{t \equiv t}$ 2. Symmetry $\dfrac{t \equiv t'}{t' \equiv t}$ 3. Transitivity $\dfrac{t \equiv t' \quad t' \equiv t''}{t \equiv t''}$
4. Substitution $\dfrac{t_1 \equiv t'_1, \ldots, t_k \equiv t'_k}{f(t_1, \ldots, t_k) \equiv f(t'_1, \ldots, t'_k)}$ for every $f \in \Sigma$ of arity k.
5. Instantiation $\dfrac{t \equiv t'}{t\rho \equiv t'\rho}$ for every substitution ρ.
6. Equations $\dfrac{}{t \equiv t'}$ for every equation $\langle t, t' \rangle \in E$

Fig. 1. The proof system $\mathbf{DED}(E)$ in [7]

We write $\vdash_E t \equiv t'$ if we can derive $t \equiv t'$; and then, we say that $t \equiv t'$ is a theorem of $\mathbf{DED}(E)$. Obviously, we can see each derivable theorem as the pair of a relation $\equiv_E$, which due to 1-3 is an equivalence relation, and as a result of 4 a Σ-congruence.

As we have said, the idea in this paper is to define distance between processes. Then, we need to extend the concept of Σ-algebra with a notion of distance. This distance allows us to measure how far away is a process p of being equivalent to another process q.

In [8] we have considered as processes the terms generated by the free $(\mathbf{0}, Act, +)$-algebra, which correspond to the classic domain of $BCCSP(Act)$ processes.

Definition 2. *Given a set of actions Act, the set BCCSP(Act) of processes is that defined by the BNF-grammar: $p ::= \mathbf{0} \mid ap \mid p + q$. The very well known operational semantics of BCCSP [10, 4] is defined by:*

$$(1)\ \frac{}{ap \xrightarrow{a} p} \qquad (2)\ \frac{p \xrightarrow{a} p'}{p + q \xrightarrow{a} p'} \qquad (3)\ \frac{q \xrightarrow{a} q'}{p + q \xrightarrow{a} q'}$$

Based on this operational semantics we can define all the semantics in the ltbt-spectrum [10]. In particular, we have studied the case of the finest of these semantics, that is the bisimulation semantics.

Definition 3 (Bisimulation). *A bisimulation between processes is a relation B such that, whenever pBq, we have:*

- *for every $a \in Act$, if $p \xrightarrow{a} p'$ there exists some q', with $q \xrightarrow{a} q'$ and $p'Bq'$.*
- *for every $a \in Act$, if $q \xrightarrow{a} q'$ there exists some p', with $p \xrightarrow{a} p'$ and $p'Bq'$.*

We have defined distances between processes corresponding to any semantics defined by a preorder $\subseteq_{\mathcal{L}}$, this covering in particular the case of equivalence relations, such as bisimulation . Both are defined in an operational way based on transformations between processes, and in a denotational way, using SOS-rules. Next, we recall this second definition

Definition 4 ([8]). *Given a semantics $\mathcal{L}$, defined by a preorder $\subseteq_{\mathcal{L}}$, coarser than bisimulation, we say that a process q is at distance at most $m \in \mathbb{N}$ of being better than some other p, w.r.t. the semantics $\mathcal{L}$ and the distance between actions $\overline{d}$, and then we write $d^{\mathcal{L}}_{\overline{d}}(p,q) \leq n$, if we can infer $p \sqsubseteq^{\mathcal{L}}_n q$, by applying the following rules:*

$$(1)\ \frac{p \subseteq_{\mathcal{L}} q}{p \sqsubseteq^{\mathcal{L}}_n q} \quad (2)\ \frac{p \sqsubseteq^{\mathcal{L}}_n q}{ap \sqsubseteq^{\mathcal{L}}_{n+\overline{d}(b,a)} bq} \quad (3)\ \frac{p \sqsubseteq^{\mathcal{L}}_n p'}{p+q \sqsubseteq^{\mathcal{L}}_n p'+q} \quad (4)\ \frac{p \sqsubseteq^{\mathcal{L}}_n q \quad q \sqsubseteq^{\mathcal{L}}_{n'} r}{p \sqsubseteq^{\mathcal{L}}_{n+n'} r}$$

For instance, for the bisimulation semantics, we have $\sim$ in the place of $\subseteq_{\mathcal{L}}$ and we simply write $=_d$ for the obtained collection of distance relations, that in this case are all symmetric.

The rest of the paper is devoted to the development of a pure algebraic framework that allows us to define those distances in an algebraic way. Once we have it, we could use all the algebraic techniques and general results from the area, on the study of those distance relations.

3 From Algebraic Semantics to Algebraic Distances

We will see in this section how the basic concepts appearing in the definition of algebraic semantics can be adequately modified in a natural way to obtain an algebraic theory of distances between processes. It can be used to get algebraic characterizations of all the distances previously presented in [8].

We start with the notion of Σ-algebra with distances, whose definition will be a resetting of the definition of quotient Σ-algebra.

Definition 5. *Let $\mathcal{D}$ be an adequate domain for distance values (e.g. $\mathbb{N}$, $\mathbb{R}^+$, $\mathbb{Q}^+$) and Σ a (classic) signature. A $(\mathcal{D},\Sigma)$-algebra is a Σ-algebra $\langle \mathcal{A}, \Sigma_{\mathcal{A}} \rangle$ and a collection of relations $\langle \equiv_d, d \in \mathcal{D} \rangle$, $\equiv_d \subseteq \mathcal{D} \times \mathcal{D}$, such that:*

1. *$a \equiv_0 a$, for all $a \in \mathcal{A}$.*
2. *$a \equiv_d b \Leftrightarrow b \equiv_d a$, for all $a, b \in \mathcal{A}$ and all $d \in \mathcal{D}$.*

3. $d \leq d'$, $a \equiv_d b \Rightarrow a \equiv_{d'} b$, *for all* $a, b \in \mathcal{A}$ *and all* $d, d' \in \mathcal{D}$.
4. $(a \equiv_d b \wedge b \equiv_{d'} c) \Rightarrow a \equiv_{d+d'} c$, *for all* $a, b, c \in \mathcal{A}$ *and all* $d, d' \in \mathcal{D}$.
5. $f \in \Sigma$, $ar(f) = k$, $a_i \equiv_{d_i} b_i$ *for all* $i \in 1..k \Rightarrow f(\overline{a}) \equiv_{\widetilde{f}(d_1,\ldots,d_k)} f(\overline{b})$, *where* $\widetilde{f}$ *is a function associated to* f *that combines the distances between the components of its arguments and satisfies* $\widetilde{f}(\overline{0}) = 0$.

Remark 1. In the previous definition we have only directly stated reflexivity of $\equiv_0$ in (1). However, applying these rules we can infer

$$(1')\ a \equiv_d a \text{ for all } a \in \mathcal{A} \text{ and all } d \in \mathcal{D},$$

by combining (1) and (3). Another possibility is to take (1′) instead of (1), and then by combining it with (4), we get the monotonicity rule (3), which therefore could be removed.

Let us discuss and justify one by one all the ingredients of Def. 5. We have introduced a collection of relations $\equiv_d$, that intuitively describe the balls of radius d of the topology induced by our distance notion. The classical properties of distances correspond to 1-4. Note how the transitivity of equivalence relations is substituted by the triangular inequality 4.

We have said that we are generalizing the notion of quotient algebra, and not just that of (plain) algebra, because we allow that $\equiv_0$ will be any Σ-congruence, and not just the equality relation. Note that in that particular case the triangular inequality becomes transitivity, because $0 + 0 = 0$, and then $\equiv_0$ has to be first an equivalence relation, but also a Σ-congruence, by applying 5.

We have preferred to be quite general w.r.t. the constraints imposed to the combination functions $\widetilde{f}$. Certainly $\widetilde{f} = +$ for all $f \in \Sigma$, will be the most interesting case. In Section 6 we will see how taking $\widetilde{f} = max$, we obtain the algebraic characterization of other noticeable distances.

Now we can get our deduction systems for distances, $\mathbf{dDED}(E_{\mathcal{D}})$, by resetting the clauses that define $\mathbf{DED}(E)$, again in a very natural way.

Definition 6. *A deduction system for distances,* ***dDED****$(E_{\mathcal{D}})$, between terms in* $\mathcal{T}_{\Sigma}(X)$*, is a collection of rules including:*

1. *Reflexivity:* $t \equiv_d t$.
2. *Symmetry:* $t \equiv_d t' \Rightarrow t' \equiv_d t$.
3. *Triangular transitivity:* $t \equiv_d t'$, $t' \equiv_{d'} t'' \Rightarrow t \equiv_{d+d'} t''$.
4. *Substitution:* $t_1 \equiv_{d_1} t'_1, \ldots, t_k \equiv_{d_k} t'_k \Rightarrow f(t_1, \ldots, t_k) \equiv_{\widetilde{f}(d_1,\ldots,d_k)} f(t'_1, \ldots, t'_k)$ *for every* $f \in \Sigma$ *of arity* k *and the corresponding* $\widetilde{f}$ *composing distances.*
5. *Instantiation:* $t \equiv_d t' \Rightarrow t\rho \equiv_d t'\rho$, *for every substitution* ρ.
6. *A set of distance equations* $E_{\mathcal{D}} = \{t_i \equiv_{d_i} t'_i \mid i \in I\}$.

Since variables are useful to get compact (possibly finite) axiomatizations, they are typically used when defining deduction systems. We have followed the same idea when defining $\mathbf{dDED}(E_{\mathcal{D}})$, that gives us distance pairs not only between

closed processes, but also between open processes. Of course, we expect that any such derivable pair will reflect a universal information, which is formalized by the instantiation rule.

The roles of reflexivity, symmetry and that of the triangular transitivity, were already commented when defining the algebras with distances. Moreover, the substitution rule states the homomorphic character of the obtained distances. Of course, we have a different deduction system for each collection of composing functions $\langle \widetilde{f} |\ f \in \Sigma \rangle$, however, we prefer to maintain this small abuse of notation.

Finally, we have again adopted quite a general point of view when defining $\mathbf{dDED}(E_{\mathcal{D}})$ based on an arbitrary set of distance equations $E_{\mathcal{D}}$. But, once more it is interesting to discuss which are the most natural sets of equations, in which we are specially interested. The role of $\mathbf{DED}(E)$ is to generate the induced set of derived equations from the set E. When we start from the axiomatization of any semantics (e.g. the bisimulation semantics), the related closed processes are exactly those having the same semantics (e.g. those bisimilar).

As explained above, $\equiv_0$ just reflects the quotient algebra on top of which we will define our distance relations. As a matter of fact the following result tells us that, when E does only contain equations on $\equiv_0$, we just obtain a system totally equivalent to an ordinary deduction system.

Proposition 1. *If $E_{\mathcal{D}}$ is a system that only contains equations on $\equiv_0$, then the system $\mathbf{dDED}(E_{\mathcal{D}})$ is "essentially" equivalent to $\mathbf{DED}(\mathrm{E})$, where $E = \{t \equiv t' \mid t \equiv_0 t' \in E_{\mathcal{D}}\}$. This means that $\vdash_{E_{\mathcal{D}}} t \equiv_d t' \Leftrightarrow \vdash_E t \equiv t',\ \forall d \in \mathcal{D}$.*

Proof. This is an immediate consequence of the following facts: (1) The triangular transitivity becomes plain transitivity when $d_1 + d_2 = 0$; (2) The impossibly to infer facts about $\equiv_0$ using other ones $\equiv_d$, with $d \neq 0$. □

Therefore, in order to have a useful deduction system for distances we need to start with a collection of equations $E_{\mathcal{D}}$ containing a set of non-trivial non-zero distance axioms $t_0 \equiv_d t_0'$ with $d > 0$. This subset is the "algebraic basis" on top of which $E_{\mathcal{D}}$ will derive the induced distance pairs in $\vdash_{E_{\mathcal{D}}} t \equiv_d t'$.

Proposition 2. *Given a system of distance equations $E_{\mathcal{D}}$, if we define $E^0_{\mathcal{D}} = \{t \equiv_d t' \mid d = 0\}$ and we consider the set of ordinary equations $E = \{t \equiv t' \mid t \equiv_0 t' \in E^0_{\mathcal{D}}\}$, then we can see the family of distance relations induced by $\vdash_{E_{\mathcal{D}}}$ as a family of distance relations between the equivalence classes induced by $\vdash_E$,*

$$[t] \equiv_d [t'] ::= \vdash_{E_{\mathcal{D}}} t \equiv_d t'$$

Proof. We only need to apply the triangular transitivity rule. □

In the following section we apply this algebraic approach, defining a distance for the bisimulation semantics.

4 An Algebraic Distance for Bisimulation

As stated above, to define our processes, we use the signature including the choice operation with arity 2, and the prefix operators $a\cdot \in Act$ with arity 1,

together with the constant null, $\mathbf{0}$. We expand this signature including variables in a set X to obtain the $BCCSP(Act, X)$ syntactic algebra. Then, the corresponding compositional approach to the definition of distances between processes in $BCCSP(Act, X)$ includes the rules:

- If we have $p \equiv_d p'$ and $q \equiv_d q'$, then $p + q \equiv_d p' + q'$.
- If we have $p \equiv_d q$, then $ap \equiv_d aq$.

Moreover, the equations characterizing the bisimulation axioms are turned into distance equations getting:

$$(B1)\ x + y \equiv_0 y + x \qquad (B2)\ x + x \equiv_0 x$$

$$(B3)\ (x + y) + z \equiv_0 x + (y + z) \qquad (B4)\ z + \mathbf{0} \equiv_0 z$$

Finally, we need to add the collection of equations that will work as seed for the computation of distances in an algebraic way. The idea is that we want to pay a tax for each punctual change. Since we are working under a function $\overline{d}$ defining the distance between actions in Act, we introduce the family of axioms with $a, b \in Act$, which can be considered as a single axiom scheme if we see a and b as generic action: $ax \equiv_{d(a,b)} bx$. Putting everything together we obtain the following algebraic definition of the bisimulation distance.

Definition 7. *Given a function distance $\overline{d}$ between the actions in Act, producing values in $\mathcal{D}$, and two processes $p, q \in BCCSP(Act, X)$, we can say that p is at most $d \geq 0$ far away of being bisimilar to q, if and only if $p \equiv_d q$ can be derived using the set of rules:*

1. *$p \equiv_d p$ for all $d \in \mathcal{D}$ and for all $p \in Proc$.*
2. *$p \equiv_d p' \Rightarrow p' \equiv_d p$ for all $d \in \mathcal{D}$ and for all $p, p' \in Proc$.*
3. *$p \equiv_{d_1} p'$ and $p' \equiv_{d_2} p'' \Rightarrow p \equiv_{d_1+d_2} p''$ for all $d_1, d_2 \in \mathcal{D}$ and $p, p', p'' \in Proc$.*
4. *(i) $p \equiv_{d_1} p'$, $q \equiv_{d_2} q' \Rightarrow p + q \equiv_{d_1+d_2} p' + q'$.*
   (ii) $p \equiv_d q \Rightarrow ap \equiv_d aq$ for all $a \in Act$, $d, d_1, d_2 \in \mathcal{D}$ and $p, p', q, q' \in Proc$.
5. *$p \equiv_d p' \Rightarrow p\rho \equiv_d p'\rho$, for every substitution ρ.*
6. *(i) $ax \equiv_{\overline{d}(a,b)} bx$ for all $a, b \in Act$.* *(ii) $x + y \equiv_0 y + x$.*
   (iii) $x + x \equiv_0 x$. *(iv) $(x + y) + z \equiv_0 x + (y + z)$.* *(v) $z + \mathbf{0} \equiv_0 z$.*

Remark 2. 1. The definition above only considers finite terms in $BCCSP(Act, X)$, but we can extend the application of these rules to infinite processes. However, this extension will only produce distances for the case of pairs of processes that are bisimilar up to the change of finitely many actions (e.g., $a^\omega \equiv_{d(a,b)} ba^\omega$). In Section 7 we will discuss how we can get other more interesting distances in the infinite case.

2. Once we use addition as the composition function of distance for the choice operator, we could substitute rule 4(i) in Def. 7 by the simpler rule $p \equiv_d p' \Rightarrow p + q \equiv_d p' + q$. We immediately obtain the original rule by combining this with the triangular transitivity rule 3.

By combining that simplified rule with rule $4(ii)$ is easy to prove that for any linear context $\mathcal{C}(x)$ we have the preservation rule

$$p \equiv_d p' \;\Rightarrow\; \mathcal{C}(p) \equiv_d \mathcal{C}(p').$$

This will not be true for any arbitrary non-linear $\mathcal{C}$, where in principle if x appears k times in $\mathcal{C}$, then we will have $\mathcal{C}(p) \equiv_{k*d} \mathcal{C}(p')$ but not $\mathcal{C}(p) \equiv_d \mathcal{C}(p')$. Obviously, this is an important difference to what happens in $\mathbf{DED}(E)$. There we can modify the global substitution rule 4 in Fig. 1 by a local substitution, where only an argument of f is substituted. It is possibly to do it without obtaining nothing new due to the presence of the transitivity rule.

Example 1. For instance, let us take $Act = \{a, b, c\}$ and define $\overline{d}(a, b) = 1$, $\overline{d}(a, c) = 2$ and $\overline{d}(b, c) = 1$. Now we will show that $ab\mathbf{0} + bb\mathbf{0} \equiv_3 ac\mathbf{0} + cc\mathbf{0}$:

$$b\mathbf{0} \equiv_1 c\mathbf{0}\ (Def.7.5,\ 7.6) \Rightarrow \left\{ \begin{array}{l} ab\mathbf{0} \equiv_1 ac\mathbf{0}\ \ (Def.7.4) \\ \qquad\wedge \\ bb\mathbf{0} \equiv_1 bc\mathbf{0}\ \ (Def.7.4) \\ \qquad\wedge \\ bc\mathbf{0} \equiv_1 cc\mathbf{0}\ \ (by\ Def.7.5,\ 7.6) \end{array} \right\} \overset{(Def.7.3)}{\Rightarrow}\ bb\mathbf{0} \equiv_2 cc\mathbf{0}$$

and finally applying Def. 7.4 we can conclude $ab\mathbf{0} + bb\mathbf{0} \equiv_3 ac\mathbf{0} + cc\mathbf{0}$.

We developed in [8] our operational and our denotational approaches to the definition of distance relations without considering variables. However we can easily extend both of them to cover the full set $BCCSP(Act, X)$. In particular, for the denotational approach that we are using here, we can extend Def. 4 by first extending the preorder $\subseteq_{\mathcal{L}}$ to open terms in the classic way, and simply applying the rest of the rules also to these open terms.

Now it is immediate to check that if we can derive $p(\overline{x}) =_d q(\overline{x})$, using this extension of Def. 4, where the variables in $\overline{x}$ are those appearing in either p or q, then we can also derive any instance $p(\overline{r}) =_d q(\overline{r})$. Here we have used the classical notation $p(\overline{r})$, to denote the application of the instantiation of each variable x_i in $p(\overline{x})$ by the corresponding term, r_i.

Lemma 1. *If we can derive $p(\overline{x}) =_d q(\overline{x})$, then we can also derive any instance of it, $p(\overline{r}) =_d q(\overline{r})$.*

Proof. By induction on the derivations of $p(\overline{x}) =_d q(\overline{x})$

1. $p(\overline{x}) \sim q(\overline{x}) \Rightarrow p(\overline{r}) \sim q(\overline{r})$, by definition of bisimulation.
2. and 3. The application of (2) and (3) in Def. 4, is immediate because instantiation satisfies the homomorphic rules:

$$(ap(\overline{r})) = a(p(\overline{r})) \text{ and } (p + q)(\overline{r}) = p(\overline{r}) + q(\overline{r}).$$

4. Finally, by a direct application of (4) in Def. 4, from $p(\overline{r}) =_n s(\overline{r})$ and $s(\overline{r}) =_{n'} q(\overline{r})$ it produces $p(\overline{r}) =_{n+n'} q(\overline{r})$. □

Next we prove the equivalence between this algebraic definition of the distance relations and the extension of the denotational definition in Def. 4.

Theorem 1. *For all p,q* $\in BCCSP(Act, X)$ *we have* $p \equiv_d q \Leftrightarrow p =_d q$

Proof. $\Leftarrow |$ We want to show that if $p =_d q$ then $p \equiv_d q$. We use induction over the depth of derivations.

1. $\frac{p \sim q}{p =_d q}$. If $p \sim q$, then we can prove it using the set of axioms for bisimulation that are mimicked by the last four axioms in Def. 7.6. So that we have $p \equiv_0 q$, and then by Def. 7.1 we have $q \equiv_d q$, and finally applying Def. 7.3 we get $p \equiv_d q$ as we wanted to show.
2. $\frac{p =_d q}{ap =_{d+\overline{d}(b,a)} bq}$. By the i.h. we have $p \equiv_d q$. We use the equation $ax \equiv_{\overline{d}(a,b)} bx$, Def. 7.1 and 7.5 to get $ap \equiv_{\overline{d}(a,b)} bp$. Moreover, applying Def. 7.4 (ii) and the i.h. we get $bp \equiv_d bq$; and finally by the triangular transitivity rule (7.3) we can conclude $ap \equiv_{d+\overline{d}(a,b)} bq$.
3. $\frac{p =_d p'}{p + q =_d p' + q}$. We have, by the i.h., that $p \equiv_d p'$. Trivially $q \equiv_0 q$ using Def. 7.1. Then, we can conclude applying 7.4 (i) that $p + q \equiv_d p' + q$.
4. $\frac{p =_d r \quad r =_{d'} q}{p =_{d+d'} q}$. Immediate, using the i.h. and Def. 7.3.

$\Rightarrow |$ By induction on the derivation of $p \equiv_d q$.

1. $p \equiv_d p$. Obviously, we have $p \sim p$ and then by Def. 4.1 we get $p =_d p$.
2. $p \equiv_d p' \Rightarrow p' \equiv_d p$. By the i.h. we get $p =_d p'$, and it is immediate to check, by induction on the derivation of $p =_d p'$ that we can generate a symmetric derivation concluding $p' =_d p$. We use that $\sim$ is an equivalence relation in Def. 4.1, and $\overline{d}$ is a symmetric relation in Def. 4.2.
3. $p \equiv_{d_1} p'$ and $p' \equiv_{d_2} p'' \Rightarrow p \equiv_{d_1+d_2} p''$. By the i.h. we get $p =_{d_1} p'$, and $p' =_{d_2} p''$, and applying Def. 4.4 we conclude $p =_{d_1+d_2} p''$.
4. (i)$p \equiv_{d_1} p'$, $q \equiv_{d_2} q' \Rightarrow p + q \equiv_{d_1+d_2} p' + q'$. By the i.h. we get $p =_{d_1} p'$ and $q =_{d_2} q'$ using Def. 4.3 and 4.4 we get $p + q =_{d_1+d_2} p' + q'$.
   (ii)$p \equiv_d q \Rightarrow ap \equiv_d aq$. Immediate, applying the i.h. and Def. 4.2.
5. $p \equiv_d p' \Rightarrow p\rho \equiv_d p'\rho$, for every substitution ρ. Trivially, applying Lemma 1.
6. (i) $ax \equiv_{\overline{d}(a,b)} bx$, applying 4.2 and 4.1.
   (ii)-(iv). Immediate, applying 4.1. □

Remark 3. Certainly, we could also remove variables from the algebraic presentation simply expanding every axiom, including instead of it, all its closed instances. Obviously the instance rules will not be necessary after that. But, of course the role of variables in any algebraic presentation is crucial in order to obtain finite axiomatizations where we reflect by a single action an infinity of facts.

5 Algebraic Distances for Other Semantics

Once we have studied in detail the algebraic definition of the distance for the case of bisimulation, we will briefly discuss the case of the rest of the semantics which are collected in the ltbt-spectrum [10]. These semantics are not induced by an equivalence relation but by a preorder. Hennessy also presented in [7] a theory of Σ-po algebras, $\langle A, \leq_A, \Sigma_A \rangle$, which are endowed with a partial order. Now for each f in Σ of arity k, there is a monotonic function f_A: $A^k \to A$. You can find in [7] all the details about this theory of ordered algebras which smoothly extends that of plain algebras. In particular, Hennessy purposes the proof system $\mathbf{DED}(I)$, in Fig. 2, where I is now a set of inequations.

1. Reflexivity $\dfrac{}{t \leq t}$
2. Transitivity $\dfrac{t \leq t' \quad t' \leq t''}{t \leq t''}$
3. Substitution $\dfrac{t_1 \leq t'_1, \ldots, t_k \leq t'_k}{f(t_1, \ldots, t_k) \leq f(t'_1, \ldots, t'_k)}$, for every $f \in \Sigma$ of arity k.
4. Instantiation $\dfrac{t \leq t'}{t\rho \leq t'\rho}$, for every substitution ρ.
5. Equations $\dfrac{}{t \leq t'}$, for every inequation $t \leq t' \in I$

Fig. 2. The proof system $\mathbf{DED}(I)$

As in Section 4 we can adapt this theory of ordered algebras, obtaining an algebraic theory which allows us to measure the distance between processes w.r.t. a given semantics $\mathcal{L}$ (in)equationally defined by axioms on an order $\subseteq^{\mathcal{L}}$. So, we define the following ordered deduction system with distances $\mathbf{dDED}(I)$.

Definition 8. *Given a semantics $\mathcal{L}$ algebraically defined by means of an axiomatization I on $\leq^{\mathcal{L}}$, and a distance $\overline{d}$ over the set of actions Act, we will say that a process p is at most $d \geq 0$ far away of being better than another process q, w.r.t. the preorder $\leq^{\mathcal{L}}$, if and only if we have $\vdash_{\boldsymbol{dDED}(I)} p \leq^{\mathcal{L}}_d q$. $\boldsymbol{dDED}(I)$ is the following deduction system:*

1. *$p \leq^{\mathcal{L}}_d p$, for all $d \in \mathcal{D}$.*
2. *$p \leq^{\mathcal{L}}_{d_1} p'$ and $p' \leq^{\mathcal{L}}_{d_2} p'' \Rightarrow p \leq^{\mathcal{L}}_{d_1+d_2} p''$.*
3. *(i) $p \leq^{\mathcal{L}}_{d_1} p'$ $q \leq^{\mathcal{L}}_{d_2} q' \Rightarrow p + q \leq^{\mathcal{L}}_{d_1+d_2} p' + q'$.*
   (ii) $p \leq^{\mathcal{L}}_d q \Rightarrow ap \leq^{\mathcal{L}}_d aq$, for all $a \in Act$.
4. *$p \leq^{\mathcal{L}}_d p'$ then $p\rho \leq^{\mathcal{L}}_d p'\rho$, for every substitution ρ.*
5. *$ax \leq^{\mathcal{L}}_{\overline{d}(a,b)} bx$.*
   $t \leq^{\mathcal{L}}_d t'$, for every inequation $t \leq^{\mathcal{L}} t' \in I$.

Remark 4. 1. By the way, the only difference between the proof systems $\mathbf{DED}(E)$ and $\mathbf{DED}(I)$ is that in this second case we are defining a preorder and not an equivalence relation, therefore symmetry is lost. The same happens in $\mathbf{dDED}(I)$ by means of which we measure "how far away" a process is to be better than another process w.r.t. the corresponding order $\subseteq^{\mathcal{L}}$.
2. Once we are defining an order and not an equivalence it is natural to consider asymmetric distances d, where $d(b,a)$ denotes what we have to add to b to obtain (at least) a. For instance if $a = 1€$ and $b = 2€$ we could have $d(a,b) = 1$ but $d(b,a) = 0$. We have developed our operational distances in [8] including this generalization, and it can be also introduced here simply changing the symmetric distance $\overline{d}$ by an asymmetric distance d.

It is easy to translate the results in Section 4 to this more general framework. In particular, we can prove that each denotational distance, obtained by application of Def.4, is equivalent to the corresponding algebraic distance, obtained by application of Def. 8.

6 Characterizing the Bisimulation Distance Game

The classic approaches to the definition of distances between processes are based on valued versions of the (bi)simulation game [9, 6, 2].

Definition 9. *(Bisimulation game) Given two LTSs, L_1 and L_2, we call configurations the pairs (p,q), with $p \in L_1$ and $q \in L_2$. The bisimulation game is played by two players: the attacker $\mathbb{A}$ and the defender $\mathbb{D}$. The initial configuration of the game deciding if $p_0 \sim q_0$, is just the pair (p_0, q_0). A round of the game, when the current configuration is (p,q), proceeds as follows:*

1. *$\mathbb{A}$ chooses either p or q.*
2. *Assuming it chooses p, he next executes a transition in L_1: $p \xrightarrow{a} p'$.*
3. *$\mathbb{D}$ must choose a transition with the same label at the other side of the board, i.e., it must execute an a-move in L_2: $q \xrightarrow{a} q'$. If $\mathbb{A}$ plays at L_2, then $\mathbb{D}$ replies at L_1.*
4. *The game proceeds in the same way from the new configuration (p', q').*

The bisimulation game can be turned into the "classical" bisimulation distance game [1], by allowing to reply any a-move by some b-move with $b \neq a$. However, in this case the defender should pay $\overline{d}(b,a)$ to the attacker for the mismatch. The value of the game, V(p,q), provides the "classical" bisimulation distance between p and q, $d_\sim(p,q)$.

In order to illustrate the generality of our algebraic approach to the definition of distances between processes, next we present an algebraic characterization of that bisimulation game distance.

Definition 10. *We define the algebraic bisimulation game collection of relations, $\{\equiv^g_d, d \in \mathcal{D}\}$, as the set of tuples $p \equiv^g_d q$ which can be derived by applying the following set of rules:*

1. $p \equiv^g_d p$ *for all* $d \in \mathcal{D}$ *and* $p \in Proc$.
2. $p \equiv^g_d p' \Rightarrow p' \equiv^g_d p$ *for all* $d \in \mathcal{D}$ *and* $p, p' \in Proc$.
3. $p \equiv^g_{d_1} p'$ *and* $p' \equiv^g_{d_2} p'' \Rightarrow p \equiv^g_{d_1+d_2} p''$ *for all* $d_1, d_2 \in \mathcal{D}$ *and* $p, p', p'' \in Proc$.
4. *(i)* $p \equiv^g_{d_1} p'$, $q \equiv^g_{d_2} q' \Rightarrow p + q \equiv^g_{max\{d_1,d_2\}} p' + q'$.
   (ii) $p \equiv^g_d q \Rightarrow ap \equiv^g_d aq$ *for all* $a \in Act$, $d, d_1, d_2 \in \mathcal{D}$ *and* $p, p', q, q' \in Proc$.
5. $p \equiv^g_d p' \Rightarrow p\rho \equiv^g_d p'\rho$, *for every substitution* ρ.
6. *(i)* $ax \equiv^g_{\overline{d}(a,b)} bx$ *for all* $a, b \in Act$. *(ii)* $x + y \equiv^g_0 y + x$.
   (iii) $x + x \equiv^g_0 x$. *(iv)* $(x + y) + z \equiv^g_0 x + (y + z)$. *(v)* $z + \mathbf{0} \equiv^g_0 z$.

Remark 5. Note that the definition of the algebraic bisimulation distance by means of the collection $\{\equiv_d, d \in \mathcal{D}\}$ in Def. 7, and that of the algebraic bisimulation game distance by means of $\{\equiv^g_d, d \in \mathcal{D}\}$, are nearly the same. We only modify the composition rule 4 (i), where the composition function $\widetilde{f}$–see Def. 5.5– was initially + and now it becomes max.

As we did for our bisimulation distance, there is still a third equivalent denotational characterization of the bisimulation game distance.

Definition 11. *We define the denotational bisimulation game collection of relations,* $\{=^g_d, d \in \mathcal{D}\}$*, as the set of tuples* $p =^g_d q$ *which can be derived by applying the following set of rules:*

$$(1)\ \frac{p \sim q}{p =^g_n q} \qquad (2)\ \frac{p =^g_d q}{ap =^g_{d+\overline{d}(b,a)} bq} \qquad (3)\ \frac{p =^g_{d_1} p'\ q =^g_{d_2} q'}{p + q =^g_{max\{d_1,d_2\}} p' + q'}$$

We have proved in [8] that the relations defined by Def. 11 remain the same if we add the transitivity rule

$$(4)\ \frac{p =^g_{d_1} q\ q =^g_{d_2} r}{p =^g_{d_1+d_2} r}$$

so further in this paper we consider Def. 11 including this rule. Then, it is easy to get the following theorem based on the proof of Th. 1

Theorem 2. $p \equiv^g_d q \Leftrightarrow p =^g_d q$.

Proof. Immediate, just substituting + by max in the reasoning related to the application of Def. 4.3 and Def. 7.4.

□

Remark 6. Although rule (3) in Def. 4 has not +, we can see applying transitivity that this rule is equivalent to

$$(3')\ \frac{p =^{\mathcal{L}}_{n_1} p' \quad q =^{\mathcal{L}}_{n_2} q'}{p + q =^{\mathcal{L}}_{n_1+n_2} p' + q'}$$

which produce a simpler proof than the given in Th. 1 when we use 4.3.

Next, we see that the original definition of the distances by means of the distance bisimulation game is also equivalent to these. We start proving two lemmas that provide us some properties of the denotational characterization and of the values of the distance bisimulation game.

Lemma 2 (Prefix lemma). *Given two processes $P = ap$ and $Q = bq$, we have $ap =^g_d bq$ if and only if there exists some d' such that $d = d' + \overline{d}(a,b)$ with $p =^g_{d'} q$.*

Proof. $\Leftarrow|$ Immediate, since if we have Def. 11.2 to $p =^g_{d'} q$ we get the result.
$\Rightarrow|$ We will prove a more general result. It says that $\sum_{i\in I} a_i p_i =^g_d \sum_{j\in J} b_j q_j$ implies $(\forall\, i \in I\, \exists\, j \in J \quad p_i =^g_{d-\overline{d}(a_i,b_j)} q_j)$. So, the result of this theorem will be just the particular case of this result when we have only one element in the sum. We use induction over the derivation of $p =^g_d q$ taking $p = \sum_{i\in I} a_i p_i$ and $q = \sum_{j\in J} b_j q_j$.

1. $\dfrac{p \sim q}{p =^g_d q}$. If $p \sim q$ then we have for each $i \in I$ such that $a_i p_i \xrightarrow{a_i} p_i$ there exists some $j \in J$ with $b_j q_j \xrightarrow{b_j} q_j$ where $b_j = a_i$ and $p_i \sim q_j$. Then we have $p_i =^g_d q_j$ and using the rule (2) we can conclude that $a_i p_i =^g_0 b_j q_j$ as we wanted to show, since $\overline{d}(a_i, b_j) = 0$.

2. $\dfrac{p =^g_{d-\overline{d}(b,a)} q}{ap =^g_{d-\overline{d}(b,a)+\overline{d}(b,a)} bq}$. Then we have a single summand at both sides, and the premise of the last set of the derivations exactly expresses the thesis to be proved.

3. If we have $p = p' + p''$, $q = q' + q''$ and $\dfrac{p' =^g_{d'} q' \quad p'' =^g_{d''} q''}{p =^g_{d=max\{d',d''\}} q}$. We will have $p' = \sum_{i\in I'} a_i p_i$, $p'' = \sum_{i\in I''} a_i p_i$, $q' = \sum_{j\in J'} b_j q_j$, and $q'' = \sum_{j\in J''} b_j q_j$ with $I = I' \cup I''$ and $J = J' \cup J''$. Then, by applying the i.h. we have $\forall i \in I'$ $\exists j \in J'$ with $p_i =^g_{d'-\overline{d}(a_i,b_j)} q_j$ and $\forall i \in I''\, \exists j \in J''$ with $p_i =^g_{d''-\overline{d}(a_i,b_j)} q_j$. As $d = max\{d', d''\}$ we immediately conclude the result applying the triangular transitivity(rule (3)).

4. $\dfrac{p =^g_d r \quad r =^g_{d'} q}{p =^g_{d+d'} q}$. We take $r = \sum_{k\in K} c_k r_k$ and $r_k =^g_{d'-\overline{d}(c_k,b_j)} q_j$, so applying the i.h. we get $\forall i \in I\ \exists k \in K\ p_i =^g_{d-\overline{d}(a_i,c_k)} r_k$ (and $\forall k \in K\ \exists j \in J$ $r_k =^g_{d'-\overline{d}(c_k,b_j)} q_j$). Then, applying the triangular inequality ($\overline{d}(a_i,b_j) \leq \overline{d}(a_i,c_k) + \overline{d}(c_k,b_j)$) we get that $\forall i \in I\ \exists j \in J$ such that $p_i =^g_{d+d'-\overline{d}(a_i,b_j)} q_j$. □

Corollary 1. *$d(\alpha,\beta) = \sum \overline{d}(a_i,b_i)$ where $\alpha = a_1 \ldots a_n$ and $\beta = b_1 \ldots b_n$. This means that we have $\alpha =^g_d \beta$ with $d = \sum_{i=1}^n \overline{d}(a_i,b_i)$, and for all $d' < d$ we have $\alpha \neq^g_{d'} \beta$.*

Proof. That $\alpha =^g_d \beta$ is immediate by iterated application of the triangular transitivity rule. We prove the second negative result by induction on n.

$\underline{n = 0|}$ We have $d = 0$ and then the result is trivial.

$\underline{n > 0|}$ Applying the Prefix Lemma –Lemma 2– we should have $\alpha' =^{g}_{d'-\overline{d}(a_1,b_1)} \beta'$ with $\alpha' = a_2 \dots a_n$ and $\beta' = b_2 \dots b_n$. But, if $d' < \sum_{i=1}^{n} \overline{d}(a_i, b_i)$ then we have $d' - \overline{d}(a_1, b_1) < \sum_{i=2}^{n} \overline{d}(a_i, b_i)$ contradicting the i.h. for the shorter sequences α' and β' □

Lemma 3. *$V(p, q) \leq V(p, r) + V(r, q)$ for all processes p,q and r.*

Proof. We use induction over the depth of processes.

For all $i \in I$ there exists some $k \in K$ with $V(p_i, r_k) = V(p, r) - \overline{d}(a_i, c_k)$. For all $k \in K$ there exists some $j \in J$ with $V(r_k, q_j) = V(r, q) - \overline{d}(c_k, b_j)$. Combining both, we obtain:

$$\text{For all } i \in I \text{ there exists some } j \in J \text{ and } k \in K \text{ with}$$
$$V(p_i, r_k) + V(r_k, q_j) = V(p, r) + V(r, q) - \overline{d}(a_i, c_k) - \overline{d}(c_k, b_j).$$

and then by applying the i.h. and the triangular inequality for d, we obtain:

$$\forall i \in I \;\; \exists j \in J \text{ with } V(p_i, q_j) \leq V(p, r) + V(r, q) - \overline{d}(a_i, b_j).$$

In a symmetric way, we prove that For all $j \in J$ there exists some $i \in I$ with $V(p_i, q_j) \leq V(p, r) + V(r, q) - \overline{d}(a_i, b_j)$. Applying the definition of the value of the distance game for bisimulation we conclude: $V(p, q) \leq V(p, r) + V(r, q)$. □

The value of the bisimulation game between two processes p and q, $V(p, q)$, gives us "the" distance between them $d_\sim(p, q)$. Next we will see that our denotational definition gives us all the bounds of this distance.

Theorem 3. *$d_\sim(p, q) = \min_d\{p =^{g}_{d} q\}$; i.e. $p =^{g}_{d'} q$ iff $d_\sim(p, q) \leq d'$.*

Proof. In order to simplify our notation we denote simply by $d(p, q)$ the distance between these two processes. We use induction on the depth of p and q.

$\underline{\supseteq |}$ We have that $p =^{g}_{v} q$ and we want to check that $d(p, q) \leq v$. We prove it by induction on the derivation of $p =^{g}_{v} q$.

1. $\dfrac{p \sim q}{p =^{g}_{v} q}$. If $p \sim q$ then the value of the bisimulation game is 0, because all along a game we can reply an a-move of p by an a-move of q, and $\overline{d}(a, a) = 0$, and conversely.

2. $\dfrac{p =^{g}_{v} q}{ap =^{g}_{v+\overline{d}(b,a)} bq}$. By applying the induction hypothesis we have $d(p, q) \leq v$. Then, the definition of the bisimulation game, produces $d(ap, bq) = d(p, q) + \overline{d}(a, b) \leq v + \overline{d}(b, a)$ as we wanted to see.

3. $\dfrac{p =^{g}_{v} q \quad p' =^{g}_{v'} q'}{p + p' =^{g}_{max\{v,v'\}} q + q'}$. By applying the induction hypothesis we have $d(p, q) \leq v$ and $d(p', q') \leq v'$. Now, any a-move on the p-side (resp. p'-side) of $p + p'$ can be replied by some b-move on the q side (resp. q'-side) of $q + q'$ guaranteeing a payment less or equal than v (resp. v'), and conversely. Thus concluding that $d(p + p', q + q') \leq max\{v, v'\}$.

4. $\dfrac{p =^g_v r \quad r =^g_{v'} q}{p =^g_{v+v'} q}$. By applying the induction hypothesis we have $d(p,r) \leq v$ and $d(r,q) \leq v'$. Now, applying Lemma 3 we conclude $d(p,q) \leq v + v'$.

$\subseteq |$ The proof is by induction on the depth of the processes. If it is 0 the result is trivial. Therefore, let us consider decomposition of the two involved processes $p = \sum a_i p_i$ and $q = \sum b_j q_j$.

Applying the definition of the bisimulation game, from $d(p,q) = v$ we obtain that for all $i \in I$ there exists some $j_i \in J$ such that $d(p_i, q_{j_i}) + \overline{d}(a_i, b_{j_i}) \leq d(p,q)$. Reciprocally, for all $j \in J$ there exists some $i_j \in I$ such that $d(p_{i_j}, q_j) + \overline{d}(a_{i_j}, b_j) \leq d(p,q)$. Now, by applying the induction hypothesis we have

$$\forall i \in I \; p_i =_{d(p_i,q_{j_i})} q_{j_i} \text{ and } \forall j \in J \; p_{i_j} =_{d(p_{i_j},q_j)} q_j.$$

From which applying Def. 11.2 we obtain $\forall i \in I \; a_i p_i =_{d(p_i,q_{i_j})+\overline{d}(a_i,b_{j_i})} b_{j_i} q_{j_i}$ and $\forall j \in J \; a_{i_j} p_{i_j} =_{d(p_{i_j},q_j)+\overline{d}(a_{i_j},b_j)} b_j q_j$, that applying the monotonicity of the bounds computed by Def. 11 produces

$$\forall i \in I \; a_i p_i =_{d(p,q)} b_{j_i} q_{j_i} \;\text{ and }\; \forall j \in J \; a_{i_j} p_{i_j} =_{d(p,q)} b_j q_j.$$

These two can be combined applying Def. 11.3 to produce $p = \sum a_i p_i =_{d(p,q)} \sum_{i \in I} b_{j_i} q_{j_i}$ and $\sum_{j \in J} a_{i_j} p_{i_j} =_{d(p,q)} \sum b_j q_j = q$.

We only need to combine these two again using Def. 11.3 and the idempotency of bisimilarity, to conclude $p =_{d(p,q)} q$, as we wanted to show. □

7 Conclusions and Future Work

We have presented an algebraic framework to define distances between processes. In particular those associated to the semantics that are axiomatizable. Although a part of our definitions and properties could be applied to arbitrary processes, most of them are based on the consideration of finite image processes. It can be syntactically represented by a (finite) term of a certain signature.

Currently we are working on the extension of our results to the infinite case. Following [7] the idea is to approximate processes by their finite approximations and compare them level by level. Then, we would state that $p \equiv_d q$ if and only if we have $p_n \equiv_d q_n \; \forall n \in \mathbb{N}$. But whenever *Act* is finite, or the non-null values of $\overline{d}(b,a)$ are low bounded, we could only obtain a finite distance $p \equiv_d q$ for some $d \in \mathcal{D}$, when q can be obtained from p by a finite number of applications of the rules in Def. 7; that is, whenever p and q are bisimilar up to a finite number of changes of the actions occurring in them.

In order to obtain a more general distance that also produces finite values for pairs of processes which cannot be transformed one into the other by a finite number of transformations, we would need to adopt a discounting function. The idea is that the weights of the disagreements between the two compared processes decrease with the depth they occur. This is easily formalized in our algebraic framework, simply changing our rule 4 *ii*) in Def. 7 by a discounted rule

$$p \equiv_d q \;\Rightarrow\; ap \equiv_{\alpha d} aq$$

where $\alpha > 1$. As a matter of fact this is another instantiation of our Def. 5. In such a case it would be immediate to check that when $\overline{d}(b, a) = 1$ and $\alpha = \frac{1}{2}$, we would have $a^\infty \equiv_2 b^\infty$.

We are also working on the definition of these distances by applying a coinductive approach that avoids the consideration of finite approximations to obtain the distances between infinite processes.

We have used the algebraic developments in [7] to base our algebraic theory on distances. We did that, not only due to the simplicity and clarity of its presentation of the theory, but also because of its detailed study of the testing semantics. We hope indeed that most, if not all, of the concepts and results on this semantics will be transferable to the distance scenario. So, we will obtain a nice theory of approximated pass of test both producing a distance for the induced semantics, and an interesting new concept to be applicable in practice.

References

[1] Černý, P., Henzinger, T.A., Radhakrishna, A.: Quantitative simulation games. In: Manna, Z., Peled, D.A. (eds.) Time for Verification. LNCS, vol. 6200, pp. 42–60. Springer, Heidelberg (2010)

[2] Černý, P., Henzinger, T.A., Radhakrishna, A.: Simulation distances. In: Gastin, P., Laroussinie, F. (eds.) CONCUR 2010. LNCS, vol. 6269, pp. 253–268. Springer, Heidelberg (2010)

[3] Chen, X., Deng, Y.: Game characterizations of process equivalences. In: Ramalingam, G. (ed.) APLAS 2008. LNCS, vol. 5356, pp. 107–121. Springer, Heidelberg (2008)

[4] de Frutos-Escrig, D., Gregorio-Rodríguez, C., Palomino, M.: On the unification of process semantics: equational semantics. ENTCS 249, 243–267 (2009)

[5] Fahrenberg, U., Legay, A., Thrane, C.R.: The quantitative linear-time–branching-time spectrum. In: FSTTCS 2011. LIPIcs, vol. 13, pp. 103–114. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2011)

[6] Fahrenberg, U., Thrane, C.R., Larsen, K.G.: Distances for weighted transition systems: Games and properties. In: QAPL 2011. EPTCS, vol. 57, pp. 134–147 (2011)

[7] Hennessy, M.: Algebraic theory of processes. MIT Press (1988)

[8] Romero Hernández, D., de Frutos Escrig, D.: Defining distances for all process semantics. In: Giese, H., Rosu, G. (eds.) FMOODS/FORTE 2012. LNCS, vol. 7273, pp. 169–185. Springer, Heidelberg (2012)

[9] Stirling, C.: Modal and temporal logics for processes. In: Moller, F., Birtwistle, G. (eds.) Logics for Concurrency. LNCS, vol. 1043, pp. 149–237. Springer, Heidelberg (1996)

[10] van Glabbeek, R.: The linear time-branching time spectrum I: the semantics of concrete, sequential processes. In: Handbook of Process Algebra, ch. 1, pp. 3–99. Elsevier (2001)

[11] Ying, M.: Topology in process calculus - approximate correctness and infinite evolution of concurrent programs. Springer (2001)

[12] Ying, M., Wirsing, M.: Approximate bisimilarity. In: Rus, T. (ed.) AMAST 2000. LNCS, vol. 1816, pp. 309–322. Springer, Heidelberg (2000)

Appendix: On the Operational Definition of the Distances between Processes

It is well known that the set of (unordered) finite trees labelled in their arcs by actions in Act: $FTree(Act)$, constitute an initial model for the theory of bisimulation. A similar result can be obtained, adding variables in X, for the set $FTree(Act, X)$ of trees which besides the constants in Act also can have variables in X labeling their leaves.

By applying Def. 7 and Prop. 2, we immediately obtain a family of distance relations on both $FTree(Act)$ and $FTree(Act, X)$. In other words these two algebras become a (D, Σ)-algebra and a $(D, \Sigma(X))$-algebra, respectively, when considering these two families of distance relations.

Next, we present in detail the corresponding operational definition, already studied in [8]. Later, we develop the proof of its equivalence with the denotational and algebraic characterizations, that were proved to be equivalent each other at Section 4.

Definition 12. *We say that an unordered tree p is at most at distance d from another tree q, w.r.t. the symmetric distance between actions $\overline{d}$, and then we write $d_{\overline{d}}(p, q) \leq d$, if and only if:*

- *(C1) $p = ap'$, $q = bp'$, and $d \geq \overline{d}(a, b)$, or*
- *(C2) $p = p' + r$, $q = q' + r$, and $d \geq d_{\overline{d}}(p', q')$, or*
- *(C3) $p = ap'$, $q = aq'$, and $d \geq d_{\overline{d}}(p', q')$, or*
- *(C4) $d \geq 0$ and q can be obtained from p by application of (B1)-(B4), or*
- *(C5) There exist r, d' and d'' s.t. $d' \geq d_{\overline{d}}(p, r)$, $d'' \geq d_{\overline{d}}(r, q)$ and $d \geq d' + d''$.*

Definition 13. *We define $p \rightsquigarrow^1_d q$ if and only if*

1. *$ap' \rightsquigarrow^1_d bp'$ and $d \geq d(a, b)$, or*
2. *$p' + r \rightsquigarrow^1_d q' + r$ and $p' \rightsquigarrow^1_d q'$, or*
3. *$ap' \rightsquigarrow^1_d aq'$ and $p' \rightsquigarrow^1_d q'$, or*
4. *$d \geq 0$ and q can be obtained from p by application of (B1)-(B4).*

Definition 14. *We define $p \rightsquigarrow_d p'$ if and only if there exist $p_1, \ldots, p_n$ such that $p \rightsquigarrow^1_{d_1} p_1 \rightsquigarrow^1_{d_2} p_2 \rightsquigarrow^1_{d_3} \cdots \rightsquigarrow^1_{d_n} p_n = p'$, where $\sum d_i = d$.*

Definition 15. *We define $p \rightsquigarrow^*_d p'$ if and only if we have $p \rightsquigarrow^1_d p'$, or there exists some p'' such that $p \rightsquigarrow^1_{d_1} p''$ and $p'' \rightsquigarrow^*_{d_2} p'$, where $d = d_1 + d_2$.*

Definition 16. *We define $p \mid\rightsquigarrow_d p'$ if and only if we have $p \rightsquigarrow^1_d p'$, or there exists some q such that $p \mid\rightsquigarrow_{d_1} q \quad q \mid\rightsquigarrow_{d_2} p'$, where $d_1 + d_2 = d$.*

Proposition 3. *Def. 14,15, 16 are obviously equivalent.*

Proof. Routine well known induction. □

Lemma 4 (Structural lemma). *If $p \rightsquigarrow_d q$ then (1) $ap \rightsquigarrow_d aq$ and (2) $p+r \rightsquigarrow_d q + r$.*

Proof. (1)| If $p \rightsquigarrow^1_d q$ then trivially we have $ap \rightsquigarrow^1_d aq$ applying Def. 13.3.
If $p \rightsquigarrow_d q$ then we use induction over the path length. Similarly, if we have $p = p_0 \rightsquigarrow^1_{d_1} p_1 \rightsquigarrow^1_{d_2} p_2 \rightsquigarrow^1_{d_3} \cdots \rightsquigarrow^1_{d_n} p_n = p'$ with $d = \sum d_i$, we will get $ap_i \rightsquigarrow^1_{d_{i+1}} ap_{i+1}$ for all $i < n$. So that, we have

$$ap = ap_0 \rightsquigarrow^1_{d_1} ap_1 \rightsquigarrow^1_{d_2} ap_2 \rightsquigarrow^1_{d_3} \cdots \rightsquigarrow^1_{d_n} ap_n = ap' \text{ with } d = \sum d_i$$

thus proving $ap \rightsquigarrow_{d_1} ap'$.

(2)| Analogous to (1). □

Theorem 4. *The operational definition of distance relations, $\rightsquigarrow_d$, and the denotational one, defining the family $=_d$, are equivalent.*

Proof. We want to prove that $p \rightsquigarrow_d q \leftrightarrow p =_d q$.

$\Rightarrow$ | We use induction over the length of the path generating $p \rightsquigarrow_d q$. We also use induction on the derivation of $p \rightsquigarrow^1_d$ to prove this basic case.

1. $p \rightsquigarrow^1_d p'$ with $p = ap''$ and $p' = bp''$ and $d \geq d(a,b)$. We only need to apply rule (2) in Def. 4 to get that $ap'' =_d bp''$.
2. $p' + r \rightsquigarrow^1_d q' + r$ with $p \rightsquigarrow^1_d p'$. The i.h. now produces that $p' =_d q'$, and then applying rule (3) in Def. 4 we get that $p' + r =_d q' + r$.
3. $ap \rightsquigarrow^1_d ap'$ with $p \rightsquigarrow^1_d p'$. One more time, the i.h. produces $p =_d p'$, and applying rule (2) in Def. 4 we get $ap =_d ap'$.
4. $p \rightsquigarrow^1_d p'$ with $d \geq 0$ and p' can be obtained from p by application of (B1)-(B4). As $p \sim p'$ trivially we have using rule (1) in Def. 4 that $p =_d p'$.
5. $p \rightsquigarrow_d p'$ if and only if $\exists p''\ p \rightsquigarrow_{d_1} p''\ p'' \rightsquigarrow_{d_2} p'$ with $d = d_1 + d_2$. We can apply the i.h. obtaining $p =_{d_1} p''$ and $p'' =_{d_2} p'$, and now applying rule (4) in Def. 4 we get $p =_{d_1+d_2} p'$, i.e., $p =_d p'$, as we wanted to show.

$\Leftarrow$ | Now we proof $p =_d q \Rightarrow p \rightsquigarrow_d q$, by induction on the derivation of $p =_d q$.

1. $\dfrac{p \sim q}{p =_d q}$. Trivially if $p \sim q$ then $p \rightsquigarrow_d q$ with $d \geq 0$, applying Def. 13.4.
2. $\dfrac{p =_{d-\overline{d}(b,a)} q}{ap =_d bq}$. By applying the induction hypothesis we have $p \rightsquigarrow_{d-\overline{d}(a,b)} q$ and applying the structural lemma we obtain $bq \rightsquigarrow_{d-\overline{d}(a,b)} bq$, from where using Def. 13.1 and Def. 15 we can conclude that $ap \rightsquigarrow_d bq$.
3. $\dfrac{p =_d p'}{p + q =_d p' + q}$. Again, by applying the induction hypothesis we have $p \rightsquigarrow_d p'$, and applying the structural lemma we conclude $p + q \rightsquigarrow_d p' + q$.
4. $\dfrac{p =_d q \quad q =_{d'} r}{p =_{d+d'} r}$. By the i.h. we have $p \rightsquigarrow_d q$ and $q \rightsquigarrow_{d'} r$. So, applying Def. 15 and Prop. 3 we get $p \rightsquigarrow_{d+d'} r$. □

chapter 7

Distances between infinite processes

"What giants? said Sancho Panza. Those thou seest there, answered his master, with the long arms, and some have them nearly two leagues long. Look, your worship, said Sancho; what we see there are not giants but windmills, and what seem to be their arms are the sails that turned by the windmake the millstone go."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter-8 Part-1

"¿Qué gigantes? dijo Sancho Panza. Aquellos que allí ves, respondió su amo, de los brazos largos, que los suelen tener algunos de casi dos leguas. Mire vuestra merced, respondió Sancho que aquellos que allí se parecen no son gigantes, sino molinos de viento, y lo que en ellos parecen brazos son las aspas, que volteadas del viento hacen andar la piedra del molino."

The Achilles heel of our research came when we tried to extend our definition of distances between processes (trees) to the infinite case. Once again, we started by considering the case of bisimulation semantics. We noticed that bisimilarity proofs could be seen as rewordings of the first level of processes (trees) assuming that the continuations have been (coinductively!) proved to be bisimilar. Using coinduction we obtained bounds for the distance between infinite, but finitary processes. Coinduction gave us a simple way to get those bounds without introducing any notion of limit. Furthermore, our coinductive definition was consistent with the original one. Concerning finite processes the distances between the "canonical" approaches of infinite processes were always smaller than the distance between the original ones. Certainly, it would be nice to prove the converse result, which would complete the proof of continuity, but at least for the moment, it has not been the case.

As we have said, the *NILS project* gave us the opportunity to met Dario Della Monica from Reykjavík University, first, during his stay in our university in the spring of 2014, and later during my own stay shared together with my supervisor, last May-June in Reykjavík. We extensively discussed in detail all our essays, and from these discussions new fresh ideas appeared. They gave us the opportunity of at least present our partial results in a satisfactory way. In particular, we included both the positive (partial) result, as well as those examples that show why it has been complicated to conclude the proof of our bisimilarity continuity theorem.

7.1 A coinductive definition for the distance between processes

This paper continued with the study of the distances between processes. We presented the coinductive definition of our distances, which extends our original ones given in the previous chapter. Our new distance seems more robust than other previous proposals. In particular, it is sound wrt the distances between the respective finite approximations, while the completeness result is still conjectured, but not yet proved.

This publication was selected to be part of the 34^{th} *IFIP International Conference on Formal Techniques for Distributed Objects, Components and Systems (FORTE 2014)* held in Berlin (Germany). The quality of this meeting is supported by the long history of this conference, which gathers a big part of the international experts from the *IFIP WG 6.1 on Architectures and Protocols for Distributed Systems*. There, I presented our results in the field of distance between processes, by showing how to extend our previous definition of distance between finite processes to the infinite case, using a nice coinductive approach.

Coinductive Definition of Distances between Processes: Beyond Bisimulation Distances*

David Romero-Hernández and David de Frutos Escrig

Dpto. Sistemas Informáticos y Computación
Facultad CC. Matemáticas, Universidad Complutense de Madrid, Spain
dromeroh@pdi.ucm.es, defrutos@sip.ucm.es

Abstract. Bisimulation captures in a coinductive way the equivalence between processes, or trees. Several authors have defined bisimulation distances based on the bisimulation game. However, this approach becomes too local: whenever we have in one of the compared processes a large collection of branches different from those of the other, only the farthest away is taken into account to define the distance. Alternatively, we have developed a more global approach to define these distances, based on the idea of how much we need to modify one of the compared processes to obtain the other. Our original definition only covered finite processes. Instead, now we present here a coinductive approach that extends our distance to infinite but finitary trees, without needing to consider any kind of approximation of infinite trees by their finite projections.

1 Introduction

Bisimulation [16,14,20] is a popular way to define the semantics of processes. Starting from their operational semantics, defined by a transition system, it captures the "natural" behavior of the processes, paying attention to the branching in them, but abstracting away from possible repetitions of equivalent behaviors. Bisimulations are just coinductive proofs of the equivalence between processes, and in fact they became one of the main causes of the popularization of the study of coinduction [20] and coalgebras [19,11,12] in the last years. They can be established in many different ways, in particular by means of the bisimulation game [21], that enlightens the co-character of bisimilarity.

When comparing two processes, the proof of their bisimilarity certainly indicates us that they are equivalent. The problem comes if we receive the information that they are not bisimilar. Then, if we substitute one component by the other, it is expected that the behavior of the full system will change "at least a bit". We want to quantify those deviations; they are formalized by our (new) distance between processes with respect to the bisimulation equivalence.

Recently, several variants of the bisimulation game have been used to define "bisimulation distances" [4,6,8,1]. They develop the seminal ideas in several previous works, such as [5,9,23]. However, as we have already illustrated in [17,18]

* Partially supported by the Spanish projects STRONGSOFT (TIN2012-39391-C04-04) and PROMETIDOS S2009/TIC-1465.

E. Ábrahám and C. Palamidessi (Eds.), FORTE 2014, LNCS 8461, pp. 249–265, 2014.

by means of several examples, these distances have some "limitations", that we try to remove by means of our new bisimulation distance. We also include in this paper some new examples enlightening the difference between our approach and those based in the bisimulation game.

Whenever we formalize the family of computations of a process we obtain a tree. Therefore, any distance between processes induces a distance between those trees. We have followed this path in the opposite way: let us look for a "natural" notion of distance between trees, and we will turn it into a distance between processes. In [17] we have presented an operational definition of our new *global bisimulation distance* for the particular case of finite trees. Roughly speaking, we define our distance between trees "computing" the costs in order to transform one of the trees into the other. We consider a given distance $\mathbf{d}$ on the alphabet of actions, so that the cost of substituting an action a by another b is given by $\mathbf{d}(a, b)$.

In this paper, we use coinduction to define the distance between processes in which we are interested. Of course, an alternative way to define the distance between infinite trees is to approximate them by their finite projections, and then taking limits. Looking for an "homogeneous" procedure that could capture all these approximations in a compact way, we introduce our coinductive distance as the coinductive "closure" of the finite transformations by means of which we defined our distance between finite processes in [17]. Once we have it, we get all the machinery of coinductive proofs in order to study our distance.

Even if the notion of tree is omnipresent in the field of semantics of processes, there is not a clearly standardized presentation of the different classes of trees in the literature. This is why we start the paper by reminding in Section 2 the definitions on trees and labelled transition systems that we use in the following. In Section 3, we recall the previous work on bisimulation distances and our alternative operational proposal covering mainly finite trees. Section 4 is the core of the paper and presents the coinductive extension of this approach covering also infinite trees. Finally, we conclude with a short section devoted to a discussion on the continuity of the coinductive distance, and the conclusions of the paper.

We strongly acknowledge the detailed reading and the suggestions of the referees of this paper, that have contributed to improve the presentation of this work.

2 Preliminaries

Let us start by recalling the coalgebraic definition of labelled transition systems (lts). As usual, we use them to represent the operational semantics of processes.

Definition 1. *Labelled Transition Systems (lts)*[1] *on a set of actions* $\mathbb{A}$ *and a set of states* N*, are given by a function* $succ : N \to \mathcal{P}(\mathbb{A} \times N)$*. We denote each*

[1] Therefore, lts's are just arc-labelled graphs, or more formally coalgebras $succ : N \to LTS(N, \mathbb{A})$ of the functor $LTS(N, \mathbb{A}) := \mathcal{P}(\mathbb{A} \times N)$ on the plain category of sets, *Set*. See for instance [12,20] for much more on coalgebras.

lts by the corresponding pair $(N, succ)$*. A lts with initial state* $(N, succ, n_0)$*, is just a lts* $(N, succ)$ *where some distinguished (initial) state* $n_0 \in N$ *is fixed. To simplify our notation, we usually remove the succ component from lts's.*

We say that any sequence $n_0 a_1 n_1 \dots a_k n_k$ with $(a_{i+1}, n_{i+1}) \in succ(n_i)\ \forall i \in \{0 \dots k-1\}$, is a path in (N, n_0). We denote the set of paths (or computations) by $Path(N, n_0)$. We say that the system N is *finite state*, if $|N| < \infty$; (N, n_0) is *finite*, if $|Path(N, n_0)| < \infty$; we say that (N, n_0) has only *finite computations*, if there is no infinite path $n_0 a_1 n_1 a_2 n_2 \dots$. We say that a system N is *finitely branching*, if for all $n \in N$ we have $|succ(n)| < \infty$.

Example 1. (See Fig.1) Two simple finite-state systems that however have infinitely many computations are the following: $N_{1,\infty} = \{n_0\}$, with $succ(n_0) = \{(a, n_0)\}$; $N_{2,\infty} = \{n_0, n_1\}$, with $succ(n_0) = succ(n_1) = \{(a, n_0), (a, n_1)\}$.

Example 2. (See Fig.1) Next three interesting non-finitely branching systems:

1. $N_{\mathbb{N}} = (\mathbb{N}, succ, 0)$ with $\mathbb{A} = \mathbb{N}$, $succ(0) = \{(k, k) \mid k \in \mathbb{N}\}$, $succ(k) = \emptyset$, $\forall k > 0$.
2. $N_2 = \{0\} \cup \{(i, j) \mid i, j \in \mathbb{N}, 1 \le j \le i\}$, taking $n_0 = 0$ with $succ(0) = \{(a, (n, 1)) \mid n \in \mathbb{N}\}$ and $succ((i, j)) = \{(a, (i, j+1))\}$ if $j < i$, while $succ((i, i)) = \emptyset$.
3. $N_2^+ = N_2 \cup \{(\infty, n) \mid n \in \mathbb{N}\}$, changing also the definition of $succ$, taking $succ(0) = \{(a, (x, 1)) \mid x \in \mathbb{N} \cup \{\infty\}\}$ and $succ((\infty, j)) = \{(a, (\infty, j+1))\}$.

We can define (rooted) trees as a particular class of lts's:

Definition 2. *We say that a system* (N, n_0) *is (or defines) a* tree *t if for all* $n \in N$ *there is a single path* $n_0 a_1 n_1 \dots a_k n_k$ *with* $n_k = n$*. Then, we say that each node* n_k *is at* level *k in t, and define* $Level_k(t) = \{n \in N \mid n \text{ is at level } k \text{ in } t\}$*. We define the* depth *of t as* $depth(t) = sup\{l \in \mathbb{N} \mid Level_l(t) \neq \emptyset\} \in \mathbb{N} \cup \{\infty\}$*. We denote by* $Trees(\mathbb{A})$ *the class of trees on the set* $\mathbb{A}$*, and by* $FTrees(\mathbb{A})$*, the subclass of finite state trees.*

Any node $n \in N$ of a tree $t = (N, succ, n_0)$ induces a subtree $t_n = (N_n, succ, n)$, where N_n is the set of nodes $n'_k \in N$ such that there exists a path $n'_0 a_1 n'_1 \dots a_k n'_k$ with $n'_0 = n$. We decompose any tree t into the formal sum $\sum_{n_{1j} \in Level_1(t)} a_j t_{n_{1j}}$. Since our trees are unordered, by definition, this formal sum is also unordered. The tree $\mathbf{0}$ corresponds to the system $(\{n_0\}, succ_0, n_0)$ with $succ_0(n_0) = \emptyset$, while if $|Level_1(t)| = 1$ we have $t = at'$, which can be reversed to define the tree at' starting from $a \in \mathbb{A}$ and $t' \in Trees(\mathbb{A})$. In a similar way, whenever $Level_1(t) = N_1 \cup N_2$ is a disjoint decomposition of that set, we can write $t = \sum_{n_{1j} \in N_1} a_j t_{n_{1j}} + \sum_{n_{1k} \in N_2} a_k t_{n_{1k}}$, which can be also reversed to define the sum $(+)$ of trees. Note that $+$ becomes commutative by definition.

For any tree $t \in Trees(\mathbb{A})$, we define its *first-level width*, that we will represent by $||t||$, as $||t|| = |Level_1(t)|$. We also define the *first k-levels width of t*, denoted by $||t||_k$, as $||t||_k = max\{||t_n|| \mid n \in \bigcup_{l \le k} Level_l(t)\}$. *Finitary trees* are just trees that are finitely branching systems, or equivalently, those such that $||t||_k < \infty$, $\forall k \in \mathbb{N}$. We denote by $FyTrees(\mathbb{A})$ the collection of *finitary trees* in $Trees(\mathbb{A})$.

All the systems in Ex.2 are indeed trees. Instead, those in Ex.1 are not trees.

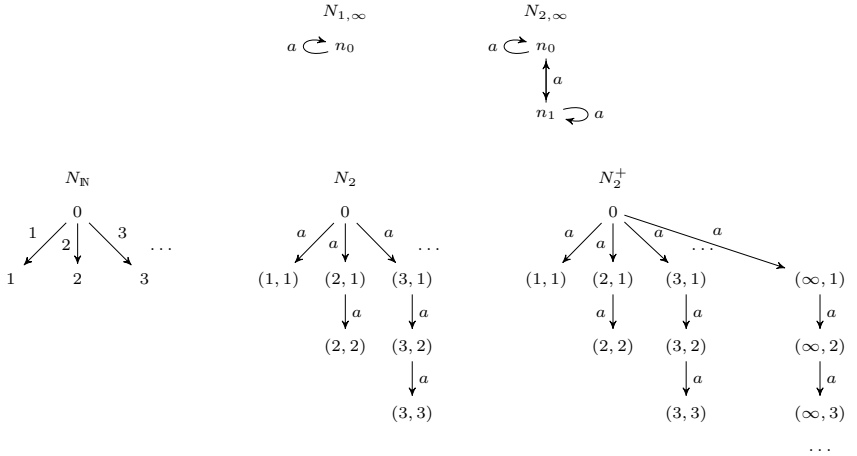


Fig. 1. Labelled Transitions Systems and Trees used in Examples 1-2

Definition 3. *Given a lts with initial state* $(N, succ, n_0)$*, we define its unfolding* $unfold(N)$ *as the tree* $(\overline{N}, \overline{succ}, \overline{n_0})$*, where* $\overline{N} = Path(N, n_0)$*,* $\overline{succ}(n_0 a_1 \dots n_k) = \{(a, n_0 a_1 \dots n_k a n') \mid (a, n') \in succ(n_k)\}$*, and* $\overline{n_0} = n_0$*.*

Definition 4. *Let* $(N, succ)$ *be a lts. We say that a relation* $\mathcal{R}$ *on* N *is a bisimulation, if for all* $(n, n') \in \mathcal{R}$ *we have*

- $\forall\ (a, n_1) \in succ(n)\ \exists\ (a, n_1') \in succ(n')$, $(n_1, n_1') \in \mathcal{R}$.
- $\forall\ (a, n_2) \in succ(n')\ \exists\ (a, n_2') \in succ(n)$, $(n_2', n_2) \in \mathcal{R}$.

We say that n *and* n' *are bisimilar if there exists some bisimulation* $\mathcal{R}$ *such that* $(n, n') \in \mathcal{R}$*, and then we write* $n \sim n'$*.*

By considering the disjoint union of two systems, we can extend the definition above to relate states from two different systems. In particular, if we consider two *lts* with initial state (or equivalently two trees) we say that $(N, succ, n_0) \sim (N', succ', n_0')$ if and only if there is a bisimulation containing the pair (n_0, n_0'). Usually we will simply write $n_0 \sim n_0'$, and the same in the case of trees, as usual.

The fact that systems are represented by their unfolding is formalized by the following result.

Proposition 1. *For any lts with initial state* $(N, succ, n_0)$*, and its unfolding* $(\overline{N}, \overline{succ}, \overline{n_0})$*, we have* $n_0 \sim \overline{n_0}$*.*

Definition 5. *Given a tree* $t = (N, succ, n_0)$ *and* $k \in \mathbb{N}$*, we define its* k*-th cut or projection,* $\pi_k(t)$*, as the restriction of* t *to the nodes in* $\bigcup_{l \le k} Level_l(t)$*:*

$$\pi_k(t) = (\pi_k(N), succ_k, n_0),\ \textit{where}\ \pi_k(N) = \textstyle\bigcup_{l \le k} Level_l(t),\ succ_k(n) = succ(n)$$
$$\textit{for}\ n \in \textstyle\bigcup_{l < k} Level_l(t),\ \textit{and}\ succ_k(n) = \emptyset\ \textit{if}\ n \in Level_k(t).$$

Proposition 2. *For any $t \in Tree(\mathbb{A})$ and $l, k \in \mathbb{N}$ with $l \leq k$, we have $\pi_l(\pi_k(t)) = \pi_l(t)$. Any finitary tree is unequivocally defined by its sequence of projections: $\forall t, t' \in FyTree(\mathbb{A})$ $(\forall k \in \mathbb{N}\ \pi_k(t) \sim \pi_k(t')) \Rightarrow t \sim t'$.*

Example 3. The result above becomes false if we consider infinitary trees. For the trees N_2 and N_2^+ in Ex.2, we have $\pi_k(N_2) \sim \pi_k(N_2^+)$ $\forall k \in \mathbb{N}$, since the "additional" branch executing a^k provided by N_2^+ can be "absorbed" by the infinitely many such branches that we already have in $\pi_k(N_2)$. Therefore, this is a (well known) counterexample disproving the continuity of bisimilarity wrt the approximations, provided by the projections π_k, if we allow infinitary trees.

As a consequence, we will restrict ourselves to finitely branching processes all along the rest of the paper. It would not be enough to consider instead just image finiteness trees, because our approach considers all the successors of each node in an homogeneous way, without taking care of their labels. Then, problems can appear as soon as a node has infinitely many successors.

3 Classical and Global Bisimulation Distances

We consider domains of actions $(\mathbb{A}, \mathbf{d})$, where $\mathbf{d} : \mathbb{A} \times \mathbb{A} \to \mathbb{R}^+ \cup \{\infty\}$ is a distance between actions, with $\mathbf{d}(a, b) = \mathbf{d}(b, a)$, $\forall a, b \in \mathbb{A}$, and, as usual, $\mathbf{d}(a, b) = 0 \Leftrightarrow a = b$, and $\mathbf{d}(a, c) + \mathbf{d}(c, b) \geq \mathbf{d}(a, b)$, $\forall a, b, c \in \mathbb{A}$ where $+$ is extended to $\mathbb{R}^+ \cup \{\infty\}$ as usual. Intuitively $\mathbf{d}(a, b) = \infty$ expresses that two actions are absolutely not interchangeable. If the value of a distance $\mathbf{d}(a, b)$ is not specified in our examples, we will assume that $\mathbf{d}(a, b) = \infty$.

The well known bisimulation game [15,22], allows us to characterize the bisimilarity relation. It is played by two players: the attacker ($\mathcal{A}$) and the defender ($\mathcal{D}$). The former executes any fireable transition from one of the compared trees, and the second has to reply it in the other tree. The attacker wins if the defender cannot counteract one of his moves; while the defender wins if he can reply forever.

Theorem 1. *([15,22]) For any t,$t' \in Trees(\mathbb{A})$ $t \sim t'$ (resp. $t \not\sim t'$) if and only if $\mathcal{D}$ (resp. $\mathcal{A}$) has a winning strategy for the bisimulation game starting at (t, t').*

Most of the recent approaches to define distances between processes –e.g. [7]– use quantitative versions of the bisimulation game. As in the plain bisimulation game, the defender has to simulate the action played by the attacker, but in this case he can fail to reply an a transition, firing instead some b. However, whenever he cheats the attacker, he has to pay him for the distance $\mathbf{d}(b, a)$. Then, the distance between two trees t and t'(equivalently, between two processes) is defined as the value of that game starting from the roots of t and t'. In the following, we will call "classical" the distances defined following this approach.

Inspired by the notion of amortized bisimulation [13], we have developed a coinductive presentation of the classical bisimulation distances [17]. Instead of

giving a definition of the distance between two trees, that requires the use of fixpoint theory, we state when an indexed family of relations between trees provides a collection of bounds on the distances between the pairs of trees in them.

As done for instance in [2,8], and thoroughly discussed in [3], when comparing pairs of processes, it is natural to introduce a "discount factor" $\alpha \in (0,1)$. Then, the differences in the k-th level of the compared trees are weighted by α^k, following the idea that differences in the far future are less important than those in the near. As a consequence, it is possible to obtain finite distances when comparing two processes with "infinitely many differences" between them. However, we will also allow that $\alpha = 1$ to cover the case in which we are not interested in the weighting of those differences.

Definition 6. *(see [17]) Given a domain of actions* $(\mathbb{A}, \mathbf{d})$ *and a discount factor* $\alpha \in (0,1]$*, we say that a family of relations between trees,* $\mathcal{R} \subseteq Trees(\mathbb{A}) \times Trees(\mathbb{A}) \times \mathbb{R}^+$*, is a classical bisimulation distance family (cbdf) for* $\mathbf{d}$ *and* α*, if it satisfies*

$$\begin{array}{ccccc} & t & R_d & t' & \\ \forall\ a\downarrow & & \Longrightarrow & \downarrow\exists b & \\ & t_1 & R_{\frac{d-\mathbf{d}(b,a)}{\alpha}} & t'_1 & \end{array} \quad \wedge \quad \begin{array}{ccccc} & t & R_d & t' & \\ \exists a\downarrow & & \Longleftarrow & \downarrow\forall\ b & \\ & t_1 & R_{\frac{d-\mathbf{d}(b,a)}{\alpha}} & t'_1 & \end{array}$$

where, we take tR_dt' *if and only if* $(t,t',d) \in \mathcal{R}$ *and implicitly, we are assuming that the values* $d - \mathbf{d}(b,a)$ *are nonnegative. We say that* t *and* t' *are at most at classical bisimulation distance* d *for the factor* α*, and then we write* $d^\alpha_{\mathbf{d}}(t,t') \leq d$*, if there is some* cbdf $\mathcal{R}$ *with* tR_dt'*.*

Proposition 3. *(see [17]) The value of the quantitative game –see [7]– defining the "classical" bisimulation distance* $dist^\alpha_{\mathbf{d}}(t,t')$ *is* $\inf(\{d \in \mathbb{R}^+ \mid d^\alpha_{\mathbf{d}}(t,t') \leq d\})$.

It is well known that, for finitary trees, this classical bisimulation distance is indeed a quantitative refinement of bisimilarity.

Theorem 2. *For all* $t, t' \in FyTrees(\mathbb{A})$ *and any discount factor* $\alpha \in (0,1]$*, we have* $t \sim t'$ *if and only if* $d^\alpha_{\mathbf{d}}(t,t') \leq 0$*, if and only if* $dist^\alpha_{\mathbf{d}}(t,t') = 0$*.*

In spite of this, we consider that in some cases this distance generates values that do not accurately reflect the differences between some pairs of trees.

Example 4. (see Fig. 2) We have a service that allows some access to the bits of our password, once we have identified ourselves in the appropriate way. Let us abstract this service as the tree $t = \sum_{i\in 1..64} a_i$. Now, let us assume that a'_i represents a cracked access to the corresponding position. Then, for each $j \in \{1 \ldots 64\}$ the system represented by the tree $t'_j = (\sum_{i\neq j} a_i) + a'_j$ certainly is wrong, but does not compromise too much the security of the system. Instead, the totally cracked system represented by $t'_{1..64} = \sum_{i\in 1..64} a'_i$ corresponds to a disastrous situation. If we take $\mathbf{d}(a_i, a'_i) = 1$, we obtain $dist^1_{\mathbf{d}}(t'_j, t) = 1$, but also $dist^1_{\mathbf{d}}(t'_{1..64}, t) = 1$.

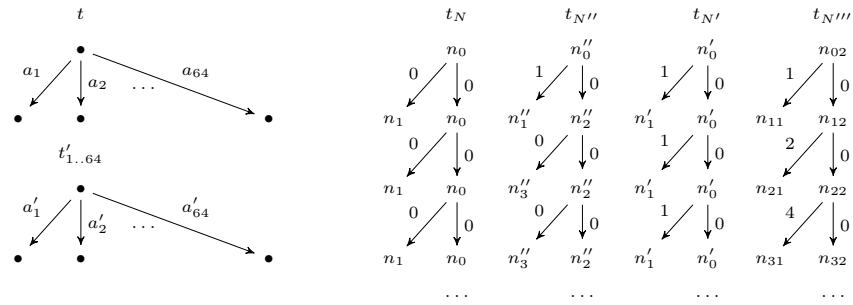


Fig. 2. Trees used in Ex. 4 - Ex.8

The bad behavior of the "classical" bisimulation distance stems from the fact that the quantitative bisimulation game only considers single computations of the compared trees. As a consequence, it cannot capture the differences "accumulated" by repeated use of a system, as illustrated in Ex.4. Later, we will see how our "global" approach copes with this feature in a more satisfactory manner.

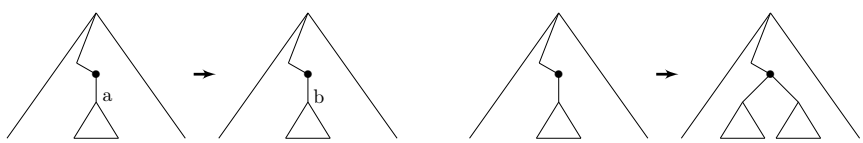
In [17], we have presented our operational definitions that allow us to obtain bounds for our new global distances between finite trees. These bounds are given by the cost of any transformation that turns one of the trees into the other. The following definition states which are the valid steps of those transformations and their costs. Roughly, any application of idempotency of + has no cost, while the change of an action a at level k into another b has as cost $\alpha^k \mathbf{d}(b,a)$.

Definition 7. *Given a domain of actions* $(\mathbb{A}, \mathbf{d})$ *and a discount factor* $\alpha \in (0,1]$*, we inductively define the distance steps on* $FTrees(\mathbb{A})$ *by*

1. $d \geq 0 \Rightarrow (t \rightsquigarrow^1_{\alpha,d} t + t \wedge t + t \rightsquigarrow^1_{\alpha,d} t)$.
2. $d \geq \mathbf{d}(a,b) \Rightarrow at \rightsquigarrow^1_{\alpha,d} bt$.
3. $t \rightsquigarrow^1_{\alpha,d} t' \Rightarrow t + t'' \rightsquigarrow^1_{\alpha,d} t' + t''$.
4. $t \rightsquigarrow^1_{\alpha,d} t' \Rightarrow at \rightsquigarrow^1_{\alpha,\alpha d} at'$.

We associate to each distance step its level, that is a natural number. The level of any step generated by 1. *or* 2. *is one; while if the level of the corresponding premise* $t \rightsquigarrow^1_{\alpha,d} t'$ *is* k*, then the level of a step generated by* 3. *(resp.* 4.*) is* k *(resp.* $k+1$*). Finally, we define the family of global distance relations* $\langle \rightsquigarrow_{\alpha,d} |\ d \in \mathbb{R}^+ \rangle$*, taking* $t \rightsquigarrow_{\alpha,d} t'$ *if there exists a sequence* $\mathcal{S} := t = t^0 \rightsquigarrow^1_{\alpha,d_1} t^1 \rightsquigarrow^1_{\alpha,d_2} t^2 \rightsquigarrow^1_{\alpha,d_3} \cdots \rightsquigarrow^1_{\alpha,d_n} t^n = t'$*, with* $\sum_{i=1}^n d_i = d$.

Therefore, we can see any sequence of distance steps turning t into t' as a sequence of local transformations $t^i \rightsquigarrow^1_{\alpha,d_i} t^{i+1}$. Each one of them either changes a single label or duplicates a partial branch at a certain level of t^i.



Remark 1. For technical reasons we want that $t \rightsquigarrow_{\alpha,d} t$ for any t, and all $d \in \mathbb{R}^+$. This can be obtained by considering the sequence $\mathcal{S} := t \rightsquigarrow^1_{\alpha,d} t + t \rightsquigarrow^1_{\alpha,0} t$.

Although at the formal level we only work with the relations $\rightsquigarrow_{\alpha,d}$, sometimes we also talk about the (global) distance defined by these bounds.

Example 5. Let us consider again the systems in Ex.4. Now, it is immediate to check that $t \rightsquigarrow_{1,1} t'_j$ for all $j \in \{1 \dots 64\}$. Therefore, in this case, the global bisimulation distance between t and any t'_j coincides with the classical bisimulation distance. However $t \rightsquigarrow_{1,64} t'_{1..64}$, but we do not have $t \rightsquigarrow_{1,d} t'_{1..64}$ for any $d < 64$: in order to transform t into $t'_{1..64}$, we need to change each a_i into a'_i, paying one unit at each step. Instead, we had $dist^1_{\mathbf{d}}(t, t'_{1..64}) = 1$. We consider that our global distance reflects in a much more accurate way the "intuitive" distance between these trees.

Example 6. We have to pass an examination about a subject with l lessons. A good student would study all of them, thus getting $S = \sum_{1..l} a_l$, which means that he totally knows the subject. At the exam the examiners choose somehow k lessons, and then each student can select a single one to develop. This means that any student that ignores up to k-1 lessons could perfectly pass the exam. These students are represented by $S_I = \sum_{i \notin I} a_i$, where I is the set of lessons that they did not study. Now, at which extend such an student is risky? What happens if the day of the exam he forgets some lesson?. If $|I| = k - 1$, then as soon as he forgets a single lesson he is in risk of failing; instead, if $|I| = 1$ he has definitely much more chances. This is again captured by our global bisimulation distance, but not by the classical one. The situation is similar to that studied in [10], where they wanted to capture how many failures are allowed before a system will fail to satisfy the requirements at its specification.

4 The Coinductive Global Bisimulation Distance

To get a general coinductive definition of our global distance for *FyTrees(𝔸)*, we keep the first three rules in Def.7, that allow us to make changes at the first level of the trees. But instead of rule 4, we introduce a coinductive rule that allows us to replace any non trivial subtree t at depth one by another t', getting a distance αd, whenever (t, t', d) is in the family that defines our global distance.

We formalize our definition in two steps. The first one, introduces the rules that produce the steps of the *coinductive transformations* between trees, starting from any family of triples (t, t', d), with t, $t' \in FyTrees(\mathbb{A})$ and $d \in \mathbb{R}^+$.

Definition 8. *Given a domain of actions $(\mathbb{A}, \mathbf{d})$, a discount factor $\alpha \in (0, 1]$ and a family $\mathcal{D} \subseteq FyTrees(\mathbb{A}) \times FyTrees(\mathbb{A}) \times \mathbb{R}^+$, we define the family of relations $\equiv^{\mathcal{D},\alpha}_d$, by:*

1. *For all $d \geq 0$ we have (i) $(\sum_{j \in J} a_j t_j) + at + at \equiv^{\mathcal{D},\alpha}_d (\sum_{j \in J} a_j t_j) + at$, and (ii) $(\sum_{j \in J} a_j t_j) + at \equiv^{\mathcal{D},\alpha}_d (\sum_{j \in J} a_j t_j) + at + at$.*
2. *$(\sum_{j \in J} a_j t_j) + at \equiv^{\mathcal{D},\alpha}_{\mathbf{d}(a,b)} (\sum_{j \in J} a_j t_j) + bt$.*

3. For all $(t, t', d) \in \mathcal{D}$ *we have* $(\sum_{j\in J} a_j t_j) + at \equiv^{\mathcal{D},\alpha}_{\alpha d} (\sum_{j\in J} a_j t_j) + at'$.

Remark 2. To simplify the notation, we will simply write $\equiv_d$ instead of $\equiv^{\mathcal{D},\alpha}_d$, whenever $\mathcal{D}$ and α will be clear from the context.

Next, the second one. Inspired by the conditions imposed to bisimulations –that can be seen as "circular proofs" of bisimilarity of all the pairs in them– we introduce the coinductive proof obligations imposed to the families of triples as above, in order to define satisfactory coinductive families of distances.

Definition 9. *Given a domain of actions* $(\mathbb{A}, \mathbf{d})$ *and a discount factor* $\alpha \in (0, 1]$*, we say that a family* $\mathcal{D}$ *is an* α*-coinductive collection of distances (*α*-ccd) between finitary trees, if for all* $(t, t', d) \in \mathcal{D}$ *there exists a* finite coinductive transformation sequence $\mathcal{C} := t = t^0 \equiv^{\mathcal{D},\alpha}_{d_1} t^1 \equiv^{\mathcal{D},\alpha}_{d_2} \ldots \equiv^{\mathcal{D},\alpha}_{d_n} t^n = t'$*, with* $d \geq \sum_{j=1}^n d_j$*. Then, when there exists an* α*-ccd* $\mathcal{D}$ *with* $(t, t', d) \in \mathcal{D}$*, we will write* $t \equiv^{\alpha}_d t'$*, and say that tree* t *is at most at distance* d *from tree* t' *wrt* α*.*

Notation: We say that the steps generated by application of rules 1 and 2 in Def.8 are *first level steps*; while those generated by rule 3 are *coinductive steps*.

Remark 3. The reason because we have introduced the condition $d \geq \sum_{j=1}^n d_j$, and not just $d = \sum_{j=1}^n d_j$, is in order to guarantee that whenever we have $t \equiv^{\alpha}_d t'$ and $d \leq d'$ we also have $t \equiv^{\alpha}_{d'} t'$. In particular, using the trivial sequence $\mathcal{C} := \mathbf{0} = \mathbf{0}$, we can prove that $\mathbf{0} \equiv^{\alpha}_d \mathbf{0}$ for all $d \in \mathbb{R}^+$. This could not be inferred if we would impose instead the condition $d = \sum_{j=1}^n d_j$. In fact, the case of $\mathbf{0}$ is the only one in which we need the inequality in Def. 9, because for any other tree t' we can apply Def. 8.1 twice, by considering any summand at of t'. Instead, in Def. 7 we can apply 7.1 even to $t = \mathbf{0}$, thus we can indeed simply take $d = \sum_{i=1}^n d_i$ at the end of the definition.

Remark 4. In order to avoid technical difficulties, the authors defining the classical bisimulation distance usually consider processes without termination. Instead, since we have mainly consider finite trees in [17], we needed to take into account termination. Our Def. 7 does not allow any "unexpected" termination when comparing two trees. If we desire to allow some terminations without necessarily entailing an infinite distance, then two simple extensions are possible. We could either establish a fixed payment f (that however will be weighted by the level at which it occurs), for any unexpected termination, including at any α-ccd $\mathcal{D}$ all the pairs $(t, \mathbf{0}, f)$, and no proof obligation for them. Or instead, we could pay for any lost action, considering a function $lost : Act \to \mathbb{R}^+$. Then, we could introduce tuples $(at + t', \mathbf{0}, d)$ in the α-ccd family $\mathcal{D}$, and for each one of them we need to check that there exist $(t, \mathbf{0}, d_1)$, $(t', \mathbf{0}, d_2) \in \mathcal{D}$ such that $\alpha d_1 + d_2 + lost(a) \leq d$. However, in order to make more understandable the paper, in the following we will not consider any of these extensions.

The next example presents a pair of trees with infinitely many differences, but a finite global bisimulation distance between them.

Example 7. (see Fig. 2) Let us consider the domain of actions $(\mathbb{N}, \mathbf{d})$, where $\mathbf{d}$ is the usual distance for numbers, and the trees $t_N = unfold(N)$ and $t_{N'} = unfold(N')$, with $N = \{n_0, n_1\}$, $succ(n_0) = \{(0, n_0), (0, n_1)\}$ and $succ(n_1) = \emptyset$; and $N' = \{n'_0, n'_1\}$, $succ'(n'_0) = \{(0, n'_0), (1, n'_1)\}$ and $succ'(n'_1) = \emptyset$. Then, we have $t_N \equiv_2^{\mathcal{D},1/2} t_{N'}$, using the family $\mathcal{D} = \{(t_N, t_{N'}, 2)\}$. We can prove that this is indeed a $\frac{1}{2}$-ccd, by considering the sequence: $\mathcal{C} := t_N \equiv_1^{\mathcal{D},1/2} t_{N''} \equiv_1^{\mathcal{D},1/2} t_{N'}$, where $t_{N''} = unfold(N'')$, with $N'' = \{n''_0, n''_1, n''_2, n''_3\}$, $succ''(n''_0) = \{(1, n''_1), (0, n''_2)\}$, $succ''(n''_1) = \emptyset$, $succ''(n''_2) = \{(0, n''_2), (0, n''_3)\}$ and $succ''(n''_3) = \emptyset$. The first step is obtained by application of rule 2 in Def.8, while the second one is obtained by application of rule 3, using the fact that $2\frac{1}{2} = 1$.

Note how the coinductive procedure "aggregates" the summands that produce the bound for the distance 2 in a single step. In fact, it is not necessary at all to sum any infinite series, as it would be the case if we would obtain that bound as the limit for the distances between the corresponding finite approximations of the two compared processes. Finally, we can observe that no bound $d < 2$ for the distance can be obtained in this way: $t_N \equiv_d^{\mathcal{D},1/2} t_{N'}$ does not hold for any $d < 2$, because for any such d we have $d < 1 + d/2$; so that, the check for the condition in Def. 9 would fail.

But, making greater the differences in the example above, we can get pairs of trees that are infinitely far away each other, wrt our global bisimulation distance.

Example 8. (see Fig. 2) Let us consider the tree t_N from Ex.7, and the tree $t_{N'''} = (N''', succ''', n_{0,2})$ with $N''' = \{n_{0,2}\} \cup \{n_{i,j} \mid i \in \mathbb{N} - \{0\}, j \in \{1, 2\}\}$, $succ'''(n_{i,1}) = \emptyset$ and $succ'''(n_{i,2}) = \{(2^i, n_{i+1,1}), (0, n_{i+1,2})\}$. We have $dist_{\mathbf{d}}^{1/2}(t_N, t_{N'''}) = 1$. Instead, $t_N \equiv_d^{1/2} t_{N'''}$ does not hold for any $d \in \mathbb{R}^+$. As a matter of fact, for the finite projections of these two trees, we have $\pi_k(t_N) \equiv_k^{1/2} \pi_k(t_{N'''})$, for all $k \in \mathbb{N}$, but we do not have $\pi_k(t_N) \equiv_d^{1/2} \pi_k(t_{N'''})$, for any $d < k$.

Based on the notion of bisimilarity, our coinductive global bisimulation distance, and the α-ccd used to define it, inherit most of its basic properties, once quantified in the adequate way.

Definition 10. *1. We say that a family $\mathcal{D}$ is triangular-transitivity closed (ttc) (resp. + closed (+c)), if for all $(t, t', d), (t', t'', d') \in \mathcal{D}$, we have $(t, t'', d+d') \in \mathcal{D}$ (resp. $(t + t'', t' + t'', d) \in \mathcal{D}$).*

2. Given a family $\mathcal{D}$, we define its tt-closure as the least family $\mathcal{D}^*$ defined by the clauses i) $\mathcal{D} \subseteq \mathcal{D}^*$; ii) If $(t, t', d), (t', t'', d') \in \mathcal{D}^*$ then $(t, t'', d + d') \in \mathcal{D}^*$.

3. Given a family $\mathcal{D}$, we define its +-closure as the family $\mathcal{D}^+ = \{(t + t'', t' + t'', d) \mid (t, t', d) \in \mathcal{D}\}$.

Proposition 4. *If $\mathcal{D}$ is an α-ccd, then $\mathcal{D}^*$ and $\mathcal{D}^+$ are too.*

As a consequence, we can assume that any ccd is ttc or +c, when convenient.

Corollary 1 (triangular-transitivity). *For any discount factor $\alpha \in (0, 1]$, whenever we have $t \equiv_d^\alpha t'$ and $t' \equiv_{d'}^\alpha t''$, we also have $t \equiv_{d+d'}^\alpha t''$.*

Next, we state the relationship between our global bisimulation distance, bisimilarity and the classical bisimulation distance.

Proposition 5. *1. For $t, t' \in FyTrees(\mathbb{A}), \alpha \in (0,1]$, we have $t \sim t' \Leftrightarrow t \equiv_0^\alpha t'$.*
2. Our global bisimulation distance is greater or equal than the classical one.

Corollary 2. *The topology induced by our global bisimulation distance is strictly finer than that induced by the classical bisimulation distance.*

Proof. It is an immediate consequence of Prop.5.2 and the (counter)Ex.8. Taking $\mathbb{R}^+$ as alphabet, and $2^i/k$ as labels of the edges of $t_{N'''}$, we obtain a family of trees $\{t^k_{N'''} \mid k \in \mathbb{N}\}$. Under the classical distance, any open ball centered in t_N, contains infinitely many trees $t^k_{N'''}$, but none of them is in any such ball for our global distance.

□

Our coinductive definition of the global bisimulation distance generalizes our operational definition for finite trees.

Lemma 1. *Any sequence $\mathcal{S}$ producing $t \rightsquigarrow_{\alpha,d} t'$ can be "factorized" into an "structured" sequence $\mathcal{T} := t = t^{0,2} \rightsquigarrow_{\alpha,d_{11}} t^{1,1} \rightsquigarrow^1_{\alpha,d_{12}} t^{1,2} \rightsquigarrow_{\alpha,d_{21}} \cdots \rightsquigarrow^1_{\alpha,d_{k2}} t^{k,2} \rightsquigarrow_{\alpha,d_{(k+1)1}} t^{k+1,1} = t'$, where the $\sum d_{1i} + \sum d_{2i} = d$, and the distance steps $t^{l,1} \rightsquigarrow^1_{\alpha,d_{l2}} t^{l,2}$ in it are exactly all the first level steps in $\mathcal{S}$. So that, no one of the subsequences producing $t^{l,2} \rightsquigarrow_{\alpha,d_{(l+1)1}} t^{l+1,1}$ contains any first level step.*

Proposition 6. *Any sequence $\mathcal{S}^l$ producing $t^{l,2} = \sum_{i=1}^m a_i t_i \rightsquigarrow_{\alpha,d_{(l+1)1}} t^{l+1,1} = \sum_{i=1}^m a_i t'_i$ can be reordered getting an "ordered" sequence $\mathcal{O} := t^{l,2} = \sum_{i=1}^m a_i t_i \rightsquigarrow_{\alpha,d^1_{(l+1)1}} \sum a_i t^1_i \rightsquigarrow_{\alpha,d^2_{(l+1)1}} \sum a_i t^2_i \rightsquigarrow_{\alpha,d^3_{(l+1)1}} \cdots \rightsquigarrow_{\alpha,d^m_{(l+1)1}} \sum a_i t^m_i = \sum_{i=1}^m a_i t'_i = t^{l+1,1}$, where $\sum_{j=1}^m d^j_{(l+1)1} = d$, $t^j_i = t'_i \ \forall j \leq i$, and $t^j_i = t_i \ \forall j > i$.*

This means that for each $j \in \{1,\ldots,m\}$ $\sum a_i t^{j-1}_i \rightsquigarrow_{\alpha,d^j_{(l+1)1}} \sum a_i t^j_i$ corresponds to $a_j t_j \rightsquigarrow_{\alpha,d^j_{(l+1)1}} a_j t'_j$, so that the distance steps in the former are exactly those from $\mathcal{S}^l$ working at the corresponding summand $a_j t_j$ of t. As a consequence, for each $j \in \{1,\ldots,m\}$ we also have $t_j \rightsquigarrow_{\alpha,(d^j_{(l+1)1})/\alpha} t'_j$, which is obtained by removing the common prefix a_j from the steps of the subsequence generating $a_j t_j \rightsquigarrow_{\alpha,d^j_{(l+1)1}} a_j t'_j$.

Proposition 7. *For $t, t' \in FTrees(\mathbb{A})$, the operational (Def.7) and the coinductive definition of our distance between trees coincide, that means $t \equiv_d^\alpha t' \Leftrightarrow t \rightsquigarrow_{\alpha,d} t'$.*

Proof. $\Rightarrow$ | Given an α-ccd relating finite trees with $(t, t', d) \in \mathcal{D}$, we can "unfold" the corresponding sequence, $\mathcal{C}$, checking $t \equiv_d^{\mathcal{D},\alpha} t'$, into a sequence of distance steps, $\mathcal{S}$, proving that $t \rightsquigarrow_{\alpha,d} t'$. We proceed by induction on $depth(t)$, as follows.

Let $t^i \equiv_{d_i}^{\mathcal{D},\alpha} t^{i+1}$ be an intermediate step in the coinductive sequence $\mathcal{C}$. If $depth(t^i) = 0$, we will trivially get $t^i \rightsquigarrow_{\alpha,d_i} t^{i+1}$. For $depth(t) \geq 1$, we apply rule 3 in Def.8, getting $t^i = t^i_1 + at_1 \equiv_{d_i}^{\mathcal{D},\alpha} t^i_1 + at'_1 = t^{i+1}$ for $(t_1, t'_1, d_i/\alpha) \in \mathcal{D}$. By

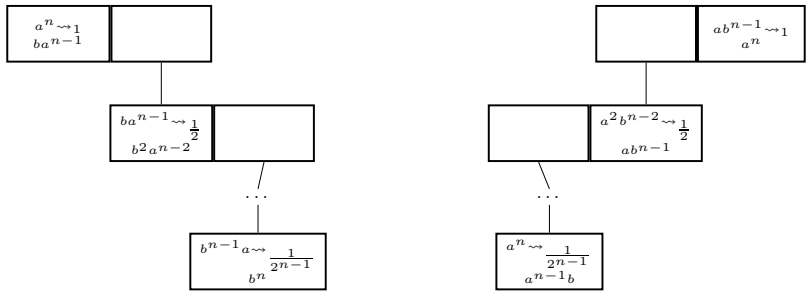


Fig. 3. Arborescent presentation of the operational sequences induced by an α-ccd

applying the induction hypothesis, we get $t_1 \rightsquigarrow_{\alpha,d_i/\alpha} t'_1$, and using rules 4 and 3 in Def.7, we obtain the desired result $t^i = t_1^i + at_1 \rightsquigarrow_{\alpha,d_i} t_1^i + at'_1 = t^{i+1}$.

$\Leftarrow|$ Given a sequence of distance steps, $\mathcal{S}$, proving that $t \rightsquigarrow_{\alpha,d} t'$, we can "fold" it into a coinductive sequence, $\mathcal{C}$, checking $t \equiv_d^{\mathcal{D},\alpha} t'$. For each $(t,t',d) \in \mathcal{D}$ we consider the factorization of the sequence $\mathcal{S}$ and its reordering as done in Prop.6. We get $t = \sum_{i\in I_0} a_i t_i \rightsquigarrow_{\alpha,d^i_{02}} \sum_{i\in I_0} a_i t'_i \rightsquigarrow^1_{\alpha,d^i_{11}} \sum_{i\in I_1} a_i t_i \rightsquigarrow_{\alpha,d^i_{12}} \sum_{i\in I_1} a_i t'_i \rightsquigarrow^1_{\alpha,d^i_{21}} \cdots \rightsquigarrow_{\alpha,d^i_{(k+1)2}} \sum_{i\in I_{k+1}} a_i t'_i = t'$, where for each sequence $\sum_{i\in I_j} a_i t_i \rightsquigarrow_{\alpha,d^i_{j2}} \sum_{i\in I_j} a_i t'_i$ and each $i \in I_j$, we have $t_i \rightsquigarrow_{\alpha,d^i_{j2}/\alpha} t'_i$ with $\sum_{i\in I_j} d^i_{j2} = d_{j2}$. Now, applying the induction hypothesis, we have $(t_i, t'_i, d^i_{j2}/\alpha) \in \mathcal{D}$, for all $i \in I_j$, so that $\sum a_i t_i \equiv^{\mathcal{D},\alpha}_{\alpha d_1} \sum a_i t^1_i \equiv^{\mathcal{D},\alpha}_{\alpha d_2} \sum a_i t^2_i \equiv^{\mathcal{D},\alpha}_{\alpha d_3} \cdots \equiv^{\mathcal{D},\alpha}_{\alpha d_{|I_j|}} \sum a_i t_i^{|I_j|} = \sum a_i t'_i$.

Therefore, each sequence $\sum_{i\in I_j} a_i t_i \rightsquigarrow_{\alpha,d^i_{j2}} \sum_{i\in I_j} a_i t'_i$ at the factorization above can be substituted by a sequence of $|I_j|$ valid coinductive steps, getting a total distance $\sum_{j=0}^{k+1} \sum_{i\in I_j} d^i_{j1} + \sum_{j=0}^{k+1} \sum_{k=1}^{|I_j|} \alpha d_k = d$.

□

Next, a pair of examples to illustrate the unfolding and folding procedures.

Example 9. Let us consider the family of trees $\{a^n \mid n \in \mathbb{N}\}$, defined by $a^0 = \mathbf{0}$ and $a^{n+1} = aa^n$. We define b^n in an analogous way. Now, if $\mathbf{d}(a,b) = 1$, we have $a^n \equiv_2^{1/2} b^n \ \forall n \in \mathbb{N}$, using $\mathcal{D} = \{(a^n, b^n, 2) \mid n \in \mathbb{N}\}$, that is shown to be a $\frac{1}{2}$-ccd by considering $\mathbf{0}$=$\mathbf{0}$ and the sequences $\mathcal{C}^n := a^n = aa^{n-1} \equiv_1^{\mathcal{D},1/2} ba^{n-1} \equiv_{1/2\cdot 2}^{\mathcal{D},1/2} bb^{n-1} = b^n$. Using the notion of unfolding above, we get the operational sequences $\mathcal{S}^n := a^n \rightsquigarrow^1_{\frac{1}{2},1} ba^{n-1} \rightsquigarrow^1_{\frac{1}{2},1/2} b^2a^{n-2} \rightsquigarrow^1_{\frac{1}{2},1/4} \cdots \rightsquigarrow^1_{\frac{1}{2},\frac{1}{2^{n-1}}} b^n$. If we preserve the structure of the sequences $\mathcal{C}^n$, whose unfolding produce these operational sequences, we can visualize them in a arborescent way –see Fig.3–. The structure reminds that of *B-trees*, where we have nodes containing keys and pointers between them. The last give access to the elements in between the former that are located at nodes at "lower" levels. By means of the (inorder) traversing of the obtained tree we recover the original operational sequences.

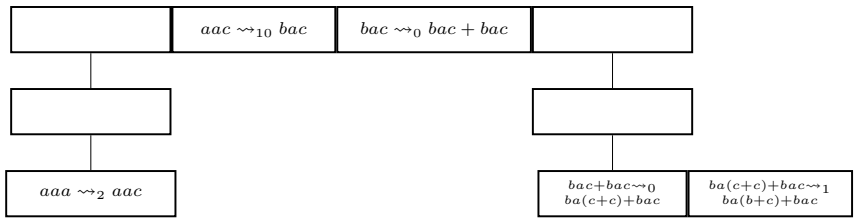


Fig. 4. Arborescent presentation of the operational sequence $\mathcal{S}$ in Ex.10

We are just "pushing the distance steps down" that correspond to "lower" levels, by introducing arcs that "move" the steps to the corresponding level. But whenever we have several steps in a row, that are not first level, then we group all of them introducing a single arc. We proceed in the same way down and down, introducing a "leaf" whenever we arrive to the level of a distance step, and new arcs going down, if there are other steps at the group at lower levels, either before or after the one which generated that leaf.

It is interesting to observe that we can also turn a^n into b^n in the opposite way, which means to use the same $\frac{1}{2}$-ccd, but a different sequence to check that it is indeed a $\frac{1}{2}$-ccd. We take now $\mathcal{C}'^n := a^n = aa^{n-1} \equiv^{\mathcal{D},1/2}_{1/2\cdot 2} ab^{n-1} \equiv^{\mathcal{D},1/2}_{1} bb^{n-1} = b^n$. Its unfolding produces the "symmetric" tree on the right of Fig.3. Certainly, you can also recognize the reversibility of the operational sequences: by reading $\mathcal{C}'^n$ from right to left we recover $\mathcal{C}^n$, simply interchanging the roles of a and b.

Example 10. Taking $\mathbb{A} = \{a, b, c\}$ with $\mathbf{d}(a, c) = 8$, $\mathbf{d}(b, c) = 4$ and $\mathbf{d}(a, b) = 10$, we obtain $aaa \rightsquigarrow^{1}_{\frac{1}{2},13} ba(b + c) + bac$, by means of the sequence $\mathcal{S} := aaa \rightsquigarrow^{1}_{\frac{1}{2},2} aac \rightsquigarrow^{1}_{\frac{1}{2},10} bac \rightsquigarrow^{1}_{\frac{1}{2},0} bac + bac \rightsquigarrow^{1}_{\frac{1}{2},0} ba(c + c) + bac \rightsquigarrow^{1}_{\frac{1}{2},1} ba(b + c) + bac$. In Fig.4 we see its arborescent presentation whose folding generates the $\frac{1}{2}$-ccd $\mathcal{D} = \{(aaa, ba(b+c)+bac, 13), (aaa, aac, 2), (aa, ac, 4), (a, c, 8), (bac+bac, ba(b+c) + bac, 1), (ac, a(b + c), 2), (c, (b + c), 4)\}$.

This is a more illustrative example of the general form of these arborescent presentations: we can have several "leaves" together with no arc in between them, when they correspond to several consecutive steps of the sequence at the current level. We can also have "degenerated" nodes, with a single arc down the tree, which corresponds to a subsequence of steps with none at the current level.

Even if Prop.7 only concerns finite trees, it reveals the duality between induction and coinduction, which is particularly interesting in the infinite case.

Example 11. Let us consider the tree $a^\infty = unfold(N_{1,\infty})$, with $N_{1,\infty}$ as in Ex.1. In an analogous way, we obtain the tree b^∞. We have $\pi_n(a^\infty) = a^n$, with a^n as in Ex.9. Therefore, a^∞ can be seen as the limit of its projections, and as we had $a^n \equiv^{1/2}_2 b^n$, we have also $a^\infty \equiv^{1/2}_2 b^\infty$. This can be proved by means of the (trivial!) collection $\mathcal{D} = \{(a^\infty, b^\infty, 2)\}$. We can check that $\mathcal{D}$ is indeed an $\frac{1}{2}$-ccd using the sequence $\mathcal{C} := a^\infty = aa^\infty \equiv^{\mathcal{D},1/2}_1 ba^\infty \equiv^{\mathcal{D},1/2}_{\frac{1}{2}\cdot 2} bb^\infty = b^\infty$.

Now, the (infinite!) "unfolding" of $\mathcal{C}$ would produce an infinite tree, that would "generate" an "infinite" operational sequence, which (intuitively) "converges" to b^∞, and "gives" us the bound 2 for the distance between a^∞ and b^∞. But our coinductive approach avoids the consideration of these limits. Moreover, the "traversing" of the arborescent presentations of the sequences, needed in many of our coinductive proofs, would produce "nested" infinite sequences much more difficult to cover without the coinductive approach.

Example 12. Let us take $\mathbb{A} = \{a, b, c, d\}$ with $\mathbf{d}(a, b) = 4$, $\mathbf{d}(c, d) = 1$. We can prove $ac^\infty + ad^\infty \equiv_6^{1/2} bc^\infty + bd^\infty$, using $\mathcal{D} = \{(ac^\infty + ad^\infty, bc^\infty + bd^\infty, 6), (c^\infty, d^\infty, 2)\}$, where the second triple in $\mathcal{D}$ is checked as in Ex.11; while for the first one we consider the coinductive sequence $\mathcal{C} := ac^\infty + ad^\infty \equiv_1^{1/2} ac^\infty + ac^\infty \equiv_0^{1/2} ac^\infty \equiv_4^{1/2} bc^\infty \equiv_0^{1/2} bc^\infty + bc^\infty \equiv_1^{1/2} bc^\infty + bd^\infty$.

Anyway, out of the informal level (where it is quite useful!) and the finite case (where it is sound), we will avoid the use of this unfolding in our formal developments. However, the following definition formalizes the use of finite unfolding, getting a generalized characterization of the relations $\equiv_d^\alpha$. It combines our two approaches (inductive, Def.7, and coinductive, Def.8,9) in a more flexible way; now operational steps can be used, not only at the first level of the trees, but also at any lower level.

Definition 11. *We consider the extension of the family of relations $\langle \rightsquigarrow^1_{\alpha,d} | \ d \in \mathbb{R}^+ \rangle$ in Def.7 to FyTrees($\mathbb{A}$). Now, given a family $\mathcal{D} = \{(t_i, t'_i, d_i) \mid i \in I\}$ with $t_i, t'_i \in FyTrees(\mathbb{A})$ and $d_i \in \mathbb{R}^+$, we define the family of relations $\widehat{\equiv}_d^{\mathcal{D},\alpha}$, by:*

1. *$t \rightsquigarrow^1_{\alpha,d} t'$ implies $t \,\widehat{\equiv}_d^{\mathcal{D},\alpha} t'$.*
2. *For all $(t, t', d) \in \mathcal{D}$ we have $(\sum_{j\in J} a_j t_j) + at \,\widehat{\equiv}_{\alpha\cdot d}^{\mathcal{D},\alpha} (\sum_{j\in J} a_j t_j) + at'$.*

Now, we can proceed exactly as in Def.9, using the relations $\widehat{\equiv}_d^{\mathcal{D},\alpha}$ instead of $\equiv_d^{\mathcal{D},\alpha}$, getting the family of relations $\widehat{\equiv}_d^\alpha$.

Proposition 8. *For all $d \in \mathbb{R}^+$, $\alpha \in (0, 1]$, the relations $\equiv_d^\alpha$ and $\widehat{\equiv}_d^\alpha$ are equal.*

The (simple) proof of this result uses the fact that operational steps not at the first level of the trees, can be "hidden" into nested coinductive steps. However, their explicit use will produce in some cases much shorter and clearer proofs.

5 On the Continuity of the Global Bisimulations Distance

We have proved in Prop.7 the consistency between our inductive and coinductive definitions for finite trees. This can be turned into the limit by considering the coinductive definition and the (finite) projections of infinite processes.

Proposition 9. *For any α-ccd $\mathcal{D}$, the projected family $\pi(\mathcal{D}) = \{(\pi_n(t), \pi_n(t'), d) \mid (t, t', d) \in \mathcal{D},\ n \in \mathbb{N}\}$ is an α-ccd that proves $t \equiv_d^\alpha t' \Rightarrow \forall n \in \mathbb{N}\ \pi_n(t) \equiv_d^\alpha \pi_n(t')$.*

Proof. Let $\mathcal{C} := t = t^0 \equiv_{d_1}^{\mathcal{D},\alpha} \ldots \equiv_{d_k}^{\mathcal{D},\alpha} t^k = t'$ be the sequence proving that $(t, t', d) \in \mathcal{D}$ satisfies the condition in order $\mathcal{D}$ to be an α-ccd. Then each projected sequence $\pi_n(\mathcal{C}) := \pi_n(t) = \pi_n(t^0) \equiv_{d_1}^{\pi(\mathcal{D}),\alpha} \ldots \equiv_{d_k}^{\pi(\mathcal{D}),\alpha} \pi_n(t^k) = \pi_n(t')$ proves that $(\pi_n(t), \pi_n(t'), d) \in \pi(\mathcal{D})$ satisfies the condition in order $\pi(\mathcal{D})$ to be an α-ccd. It is clear that the projection under π_n of any first level step in $\mathcal{C}$, is also a valid step in $\pi_n(\mathcal{C})$. Moreover, any coinductive step in $\mathcal{C}$ using $(t_1, t_1', d) \in \mathcal{D}$, can be substituted by the corresponding projected step, that uses $(\pi_{n-1}(t_1), \pi_{n-1}(t_1'), d) \in \pi(\mathcal{D})$. □

Remark 5. Alternatively, we can consider for each $n \in \mathbb{N}$ a family $\mathcal{D}_n = \pi_n(\mathcal{D}) = \{(\pi_m(t), \pi_m(t'), d) \mid (t, t', d) \in \mathcal{D},\ m \in \mathbb{N} \wedge m \leq n\}$, using the fact that the subtrees of a projection $\pi_n(t)$ are also projections $\pi_m(t'')$ of subtrees t'' of t, for some $m < n$. These families satisfy $\pi_m(\mathcal{D}) \subseteq \pi_n(\mathcal{D})$, whenever $m \leq n$.

Example 13. Let us consider the trees a^∞ and b^∞ in Ex.11 and the $\frac{1}{2}$-ccd $\mathcal{D} = \{(a^\infty, b^\infty, 2)\}$ that proves $a^\infty \equiv_2^{1/2} b^\infty$, by means of the sequence $\mathcal{C} := a^\infty = aa^\infty \equiv_1^{\mathcal{D},1/2} ba^\infty \equiv_{\frac{1}{2}\cdot 2}^{\mathcal{D},1/2} bb^\infty = b^\infty$. Now for the families $\mathcal{D}_n = \pi_n(\mathcal{D})$ in Remark 5, we have $\mathcal{D}_n = \{(a^m, b^m, 2) \mid m \leq n\}$, which gives us $a^n \equiv_2^{1/2} b^n$ by means of the sequence $\mathcal{C}^n = \pi_n(\mathcal{C}) := a^n = aa^{n-1} \equiv_1^{\mathcal{D}_n,1/2} ba^{n-1} \equiv_{\frac{1}{2}\cdot 2}^{\mathcal{D}_n,1/2} bb^{n-1} = b^n$.

We conjecture that the converse of Prop.9 asserting the continuity of our coinductive distance, is also true. Unfortunately, the proof of this result is being much more complicated than we expected. Our idea, is to use the reasoning in the proof of Prop.9 in the opposite direction and the correspondence between operational and coinductive sequences in the finite case. As far as we have a collection of "uniform"[2] operational sequences $\mathcal{S}^n := \pi_n(t) \rightsquigarrow_{\alpha,d} \pi_n(t')$, we could "overlap" all of them getting an infinite tree as that in Fig.3. By "folding" this tree we obtain the coinductive sequence $\mathcal{C}$, proving $t \equiv_d^\alpha t'$. Next we provide a simple example.

Example 14. Let us consider the trees $t = ac^\infty + ad^\infty$ and $t' = bc^\infty + bd^\infty$, as in Ex.12, and the same distance $\mathbf{d}$ as there. Then we have:

$$\pi_1(t) = a + a \rightsquigarrow_{\frac{1}{2},0}^{(1)} a \rightsquigarrow_{\frac{1}{2},4}^{(1)} b \rightsquigarrow_{\frac{1}{2},0}^{(1)} b + b = \pi_1(t'),$$

$$\pi_2(t) = ac+ad \rightsquigarrow_{\frac{1}{2},\frac{1}{2}}^{(2)} ac+ac \rightsquigarrow_{\frac{1}{2},0}^{(1)} ac \rightsquigarrow_{\frac{1}{2},4}^{(1)} bc \rightsquigarrow_{\frac{1}{2},0}^{(1)} bc+bc \rightsquigarrow_{\frac{1}{2},\frac{1}{2}}^{(2)} bc+bd = \pi_2(t'),$$

$$\pi_3(t) = acc + add \rightsquigarrow_{\frac{1}{2},\frac{1}{2}\cdot 1}^{(2)} acc + acd \rightsquigarrow_{\frac{1}{2},\frac{1}{4}\cdot 1}^{(3)} acc + acc \rightsquigarrow_{\frac{1}{2},0}^{(1)} acc \rightsquigarrow_{\frac{1}{2},4}^{(1)}$$
$$bcc \rightsquigarrow_{\frac{1}{2},0}^{(1)} bcc + bcc \rightsquigarrow_{\frac{1}{2},\frac{1}{2}\cdot 1}^{(2)} bcc + bdc \rightsquigarrow_{\frac{1}{2},\frac{1}{4}\cdot 1}^{(3)} bcc + bdd = \pi_3(t').$$

We have included the superscripts (k) to indicate at which level we apply each transformation step. Each of these sequences can be obtained from the following one by removing the steps marked with $(i+1)$ and applying π_i.

Now, if we consider the operational sequences, $\mathcal{S}^n$, relating $\pi_n(t)$ and $\pi_n(t')$, for any $n \in \mathbb{N}$, we obtain $\pi_n(t) \rightsquigarrow_{\frac{1}{2},d_n} \pi_n(t')$, for some $d_n < 6$. For instance, we

[2] Uniformity here means that for any $n, k \in \mathbb{N}$ with $k \geq n$ the steps of all the sequences $\mathcal{S}^k$ corresponding to the first n-levels are always the same.

get $\pi_1(t) \rightsquigarrow_{\frac{1}{2},4} \pi_1(t')$, $\pi_2(t) \rightsquigarrow_{\frac{1}{2},5} \pi_2(t')$ and $\pi_3(t) \rightsquigarrow_{\frac{1}{2},5.5} \pi_3(t')$. The obtained distances form an increasing (but bounded) sequence, since each of the operational sequences expand the former ones, adding new costs caused by the (new) differences at the bottom levels.

Turning these operational sequences into coinductive ones, $\mathcal{C}^n$, as in Prop.7 we obtain the proof of $\pi_n(t) \equiv_6^{1/2} \pi_n(t')$, for all $n \in \mathbb{N}$. Here, it is convenient to use the (same) value 6 at all the cases.

$$\mathcal{C}^1 := a + a \equiv_1^{1/2} a + a \equiv_0^{1/2} a \equiv_4^{1/2} b \equiv_0^{1/2} b + b \equiv_1^{1/2} b + b,$$
$$\mathcal{C}^2 := ac + ad \equiv_1^{1/2} ac + ac \equiv_0^{1/2} ac \equiv_4^{1/2} bc \equiv_0^{1/2} bc + bc \equiv_1^{1/2} bc + bd,$$
$$\mathcal{C}^3 := acc + add \equiv_1^{1/2} acc + acc \equiv_0^{1/2} acc \equiv_4^{1/2} bcc \equiv_0^{1/2} bcc + bcc \equiv_1^{1/2} bcc + bdd.$$

We expect that whenever we have $\pi_n(t) \equiv_d^{\alpha} \pi_n(t')\ \forall n \in \mathbb{N}$ there will be a collection of uniform sequences proving these facts. For $t, t' \in FTrees(\mathbb{A})$ with $||t||_n, ||t'||_n \leq l$ and $t \rightsquigarrow_{\alpha,d} t'$, we should prove this by means of a sequence $\mathcal{S}$ that only uses intermediate trees t'' with $||t''||_n \leq f(l, n)$, for a certain function f. But, the existence of such a uniform bound is still to be proved.

6 Conclusions and Future Work

We have presented a coinductive characterization of our global bisimulation distance, that previously we presented in an operational and an algebraic way. So, we extend our distance to the case of infinite trees without needing to introduce any complex notion of limit of our finite transformations generating the distances between finite trees. The coinductive approach makes the work in a much easier way. Our coinductive distances are always "sound" wrt the distances between their respective finite approximations. We expect that the "completeness" result, ending the proof of continuity, will also be true.

Besides the work devoted to complete the proof of the continuity theorem, now we are working in two complementary directions. On the one hand, we will try to apply our coinductive distance in order to define distances for testing, which should state how far away is a process to pass the tests imposed by any specification. On the other hand, we will continue the theoretical study of our coinductive distances. We consider that the results here are very promising, showing a new field of application of coinductive techniques into the study of the semantics of processes. We hope that much more will be shortly coming.

References

1. Černý, P., Henzinger, T.A., Radhakrishna, A.: Quantitative simulation games. In: Manna, Z., Peled, D.A. (eds.) Time for Verification. LNCS, vol. 6200, pp. 42–60. Springer, Heidelberg (2010)
2. Černý, P., Henzinger, T.A., Radhakrishna, A.: Simulation distances. In: Gastin, P., Laroussinie, F. (eds.) CONCUR 2010. LNCS, vol. 6269, pp. 253–268. Springer, Heidelberg (2010)

3. de Alfaro, L., Henzinger, T.A., Majumdar, R.: Discounting the future in systems theory. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, pp. 1022–1037. Springer, Heidelberg (2003)
4. de Alfaro, L., Majumdar, R., Raman, V., Stoelinga, M.: Game relations and metrics. In: LICS 2007, pp. 99–108. IEEE Computer Society (2007)
5. Desharnais, J., Gupta, V., Jagadeesan, R., Panangaden, P.: Metrics for labeled markov systems. In: Baeten, J.C.M., Mauw, S. (eds.) CONCUR 1999. LNCS, vol. 1664, pp. 258–273. Springer, Heidelberg (1999)
6. Desharnais, J., Laviolette, F., Tracol, M.: Approximate analysis of probabilistic processes: Logic, simulation and games. In: QEST 2008, pp. 264–273. IEEE Computer Society (2008)
7. Fahrenberg, U., Legay, A., Thrane, C.R.: The quantitative linear-time–branching-time spectrum. In: FSTTCS 2011. LIPIcs, vol. 13, pp. 103–114. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2011)
8. Fahrenberg, U., Thrane, C.R., Larsen, K.G.: Distances for weighted transition systems: Games and properties. In: QAPL 2011, pp. 134–147 (2011)
9. Giacalone, A., Jou, C., Smolka, S.A.: Algebraic reasoning for probabilistic concurrent systems. In: Proc. IFIP TC2 Working Conference on Programming Concepts and Methods, pp. 443–458. North-Holland (1990)
10. Henzinger, T.A., Otop, J.: From model checking to model measuring. In: D'Argenio, P.R., Melgratti, H. (eds.) CONCUR 2013 – Concurrency Theory. LNCS, vol. 8052, pp. 273–287. Springer, Heidelberg (2013)
11. Jacobs, B.: Exercises in coalgebraic specification. In: Blackhouse, R., Crole, R.L., Gibbons, J. (eds.) Algebraic and Coalgebraic Methods in the Mathematics of Program Construction. LNCS, vol. 2297, pp. 237–280. Springer, Heidelberg (2002)
12. Jacobs, B.: Introduction to coalgebra. towards mathematics of states and observations (2012), `http://www.cs.ru.nl/B.Jacobs/CLG/JacobsCoalgebraIntro.pdf`
13. Kiehn, A., Arun-Kumar, S.: Amortised bisimulations. In: Wang, F. (ed.) FORTE 2005. LNCS, vol. 3731, pp. 320–334. Springer, Heidelberg (2005)
14. Milner, R.: Communication and concurrency. Prentice Hall (1989)
15. Nielsen, M., Clausen, C.: Bisimulation, games, and logic. In: Karhumäki, J., Rozenberg, G., Maurer, H.A. (eds.) Results and Trends in Theoretical Computer Science. LNCS, vol. 812, pp. 289–306. Springer, Heidelberg (1994)
16. Park, D.M.R.: Concurrency and automata on infinite sequences. In: Deussen, P. (ed.) GI-TCS 1981. LNCS, vol. 104, pp. 167–183. Springer, Heidelberg (1981)
17. Romero Hernández, D., de Frutos Escrig, D.: Defining distances for all process semantics. In: Giese, H., Rosu, G. (eds.) FMOODS/ FORTE 2012. LNCS, vol. 7273, pp. 169–185. Springer, Heidelberg (2012)
18. Romero Hernández, D., de Frutos Escrig, D.: Distances between processes: A pure algebraic approach. In: Martí-Oliet, N., Palomino, M. (eds.) WADT 2012. LNCS, vol. 7841, pp. 265–282. Springer, Heidelberg (2013)
19. Rutten, J.J.M.M.: Universal coalgebra: a theory of systems. Theor. Comput. Sci. 249(1), 3–80 (2000)
20. Sangiorgi, D.: Advanced topics in bisimulation and coinduction. Cambridge Tracts in Theoretical Computer Science (2011)
21. Stirling, C.: The joys of bisimulation. In: Brim, L., Gruska, J., Zlatuška, J. (eds.) MFCS 1998. LNCS, vol. 1450, pp. 142–151. Springer, Heidelberg (1998)
22. Stirling, C.: Bisimulation, modal logic and model checking games. Logic Journal of the IGPL 7(1), 103–124 (1999)
23. Ying, M., Wirsing, M.: Approximate bisimilarity. In: Rus, T. (ed.) AMAST 2000. LNCS, vol. 1816, pp. 309–322. Springer, Heidelberg (2000)

7.2 Some ideas for proving the continuity

The last publication included in this thesis presents our partial results about the continuity of our coinductive distance between infinite processes. When addressing this result, we expected to obtain a (relatively) simple proof by showing that the overlapping of the transformations proving the common bound for the distance between the (corresponding) projections of the compared processes, would produce a coinductive transformation, proving the same bound between the original infinite processes. This was mainly because we had indeed proved the result for the two first levels of the transformation and then we expected to be able to combine these results to produce the whole finite transformation. Unfortunately all the inductive reasonings by means of which we tried to do it have been failing one after the others ....

Our paper titled *Proving Continuity of Coinductive Global Bisimulation Distances: A Never ending Story* has been presented in the *XV Jornadas de Programación y Lenguajes (PROLE 2015)* held in Santander (Spain). We have been invited to send a revised version that we expect it will be accepted for its publication in the proceedings of the workshop, to appear in the volume of the *Electronic Proceedings in Theoretical Computer Science* series, dedicated to this event.

Proving Continuity of Coinductive Global Bisimulation Distances: A Never Ending Story*

David Romero-Hernández [1] David de Frutos-Escrig [2] Dario Della Monica [3]

[1,2] Facultad CC. Matemáticas, Universidad Complutense de Madrid Madrid, Spain.
Departamento de Sistemas Informáticos y Computación

[3] ICE-TCS, School of Computer Science, Reykjavik University, Reykjavik, Iceland

dromeroh@pdi.ucm.es defrutos@sip.ucm.es dariodm@ru.is

We have developed a notion of global bisimulation distance between processes which goes somehow beyond the notions of bisimulation distance already existing in the literature, mainly based on bisimulation games. Our proposal is based on the cost of transformations: how much we need to modify one of the compared processes to obtain the other. Our original definition only covered finite processes, but a coinductive approach allows us to extend it to cover infinite but finitary trees. After having shown many interesting properties of our distance, it was our intention to prove continuity with respect to projections, but unfortunately the issue remains open. Nonetheless, we have obtained several partial results that are presented in this paper.

1 Introduction

The notion of bisimulation has been extensively used to characterize the equivalence between processes [12, 13, 18]. Bisimulations are coinductive proofs of that equivalence, which is called the bisimilarity relation. Certainly, bisimilarity is a quite natural relation, as suggested by the existence of several different formulation of the notion of bisimulation, e.g., in terms of bisimulation games [19].

Up to bisimilarity, the semantics of processes is characterized by unordered trees without repeated (equivalent) branches, which are thus considered the canonical semantic model. Therefore, two processes are equivalent if, and only if, they have the same semantic tree. But when two processes are not equivalent we have no way of expressing "how different" they are. Recently, several notions of bisimulation distance have been proposed, based on variants of the bisimulation game [1, 4, 7, 8]: while the original game imposes to the defender the obligation of replicating exactly any move by the attacker, in these variants the defender has the possibility of "cheating", by replying an attacker's move by choosing similar, but not equal, actions. However, when doing that, the defender have to pay a price according to the distance between the two involved actions.

This is a very suggestive path to follow when defining a bisimulation distance, and it comes with several efficient ways to compute it. Despite such desirable properties, we believe that alternative approaches are possible and worth being studied. Previous works [15, 16, 17] by the first two authors of this paper contain several examples which mainly show that the "classical" distances based on variants

*This work was Partially supported by [1,2] The Spanish project STRONGSOFT (TIN2012-39391-C04-04) and UCM-Santander grant GR3/14.
[2] The Spanish project N-GREENS Software-CM (S2013/ICE-2731).
[3] The Icelandic project *Processes and Modal Logics* (project nr. 100048021) and *Decidability and Expressiveness for Interval Temporal Logics* (project nr. 130802-051) of the Icelandic Research Fund.
[1,2,3] The EEA Grants from the project *Formal Methods for the Development and Evaluation of Sustainable Systems* (001-ABEL-CM-2013).

Submitted to:
PROLE 2015

of the bisimulation game are *local*, in the sense that they only capture the difference between a single pair of executions of the two processes, thus failing in characterizing the distance between the processes in their entirety.

In the quest for a *global* notion of distance, considering all executions at the same time, we have proposed a novel approach [15, 16, 17]: since trees express the bisimulation semantics, we looked for a natural distance between trees that defines what we called *global bisimulation distance*. For this purpose, we defined *atomic transformations* between processes; then, a sequence of transformations provides an upper bound for the distance between the two processes at the two ends of the sequence.

This approach is suitable for comparing finite processes, but it is clearly inadequate when comparing infinite finitary processes with infinitely many differences. We are interested in obtaining sound bounds also in the latter case, whenever the series collecting all those differences converges. Instead of looking for a complex scenario based on the use of limits, we introduced in [17] a coinductive framework which allows us to obtain bounds for those distances in a very simple way (coinduction is sometimes presented as an "inductionless induction" mechanism).

As we said before, classical bisimulation distances are easy to calculate, even (or we should better say, especially) in the quantitative cases (e.g., probabilistic [21], timed [20]), where calculus provides the machinery to obtain the corresponding fixed points. It is true that the computation of our (bounds for the) distance requires specific techniques in each case, but our coinductive approach benefits from the power of coinduction to accomplish this task.

In order to give a broader support to our approach, it was our intention to prove the continuity of our distance: we expected that whenever all the pairs of projections of two (possibly infinite) processes are at some fixed distance, the (full) processes themselves will be at that distance. Unfortunately, this paper tells an unfinished story: we were confident about obtaining a perhaps bit involved, but somehow "standard" proof, but our creature has revealed itself as an irresistible beast. Our, more and more, sophisticated attempts to domesticate it have crashed over and over, revealing new faces of the beast, whose actual existence remains uncertain. We all know how difficult is to disprove the existence of Nessy, Bigfoot, or E.T. Certainly, finding indisputable evidence of their existence would put an end to the mystery, but still we believe that this is impossible, as they simply do not exist ... or do they?

Let us go with our story. It is not easy to tell unfinished stories, but we think that we have to do it, because we really enjoyed many exciting adventures in pursuing our quest and because we feel we might have paved the way for somebody else's success.

2 Classic and global bisimulation distances

Our starting point will be the operational definition of processes as *Labelled Transition Systems (lts)* over an alphabet $\mathbb{A}$, which are tuples $(N, succ)$, where N is a set of states and $succ : N \to \mathscr{P}(\mathbb{A} \times N)$. When we want to distinguish an initial state $n_0 \in N$, we write $(N, succ, n_0)$. Sometimes we omit the component *succ*, simply considering (N, n_0). Finite computations or *paths*, are sequences $n_0 a_1 n_1 \dots a_k n_k$ with $(a_{i+1}, n_{i+1}) \in succ(n_i)\ \forall i \in \{0 \dots k-1\}$. We denote the set of paths of an *lts* (N, n_0) by $Path(N, n_0)$.

Definition 1. *We say that an lts* (N, n_0) *is (or defines) a* tree *t if for all* $n \in N$ *there is a single path* $n_0 a_1 n_1 \dots a_j n_j$ *with* $n_j = n$*. Then, we say that each node* n_k *is at* level *k in t, and define* $Level_k(t) = \{n \in N \mid n \text{ is at level } k \text{ in } t\}$*. We define the* depth *of t as* $depth(t) = sup\{l \in \mathbb{N} \mid Level_l(t) \neq \emptyset\} \in \mathbb{N} \cup \{\infty\}$*. We denote by* $Trees(\mathbb{A})$ *the class of trees on the alphabet* $\mathbb{A}$*, and by* $FTrees(\mathbb{A})$*, the subclass of finite trees.*

Every node n of a tree $t = (N, n_0)$ induces a subtree $t_n = (N_n, n)$, where N_n is the set of nodes $n'_k \in N$ such that there exists a path $n'_0 a_1 n'_1 \dots a_k n'_k$ with $n'_0 = n$. We decompose any tree t into the formal sum

$\sum_{n_{1j}\in Level_1(t)} a_j t_{n_{1j}}$ (we denote the empty sum by $\mathbf{0}$). Since our trees are unordered, by definition, this formal sum is also unordered.

For each tree $t \in Trees(\mathbb{A})$, we define its *first-level width*, that we represent by $||t||$, as $||t|| = |Level_1(t)|$. We also define the *first k-levels width of t*, denoted by $||t||_k$, as $||t||_k = max\{||t_n|| \mid n \in \bigcup_{l<k} Level_l(t)\}$. *Finitary trees* are those such that $||t||_k < \infty$, $\forall k \in \mathbb{N}$. We denote by $FyTrees(\mathbb{A})$ the collection of *finitary trees* in $Trees(\mathbb{A})$.

Definition 2. *Given an lts with initial state $(N, succ, n_0)$, we define its unfolding, denoted by $unfold(N)$, as the tree $(\overline{N}, \overline{succ}, \overline{n_0})$, where $\overline{N} = Path(N, n_0)$, $\overline{succ}(n_0 a_1 \dots n_k) = \{(a, n_0 a_1 \dots n_k a n') \mid (a, n') \in succ(n_k)\}$, and $\overline{n_0} = n_0$. An lts is finitely branching when its unfolding is a finitary tree.*

Definition 3. *Given a tree $t = (N, succ, n_0)$ and $k \in \mathbb{N}$, we define its k-th cut or projection, denoted by $\pi_k(t)$, as the restriction of t to the nodes in $\bigcup_{l\le k} Level_l(t)$:*

$$\pi_k(t) = (\pi_k(N), succ_k, n_0), \text{ where } \pi_k(N) = \bigcup_{l\le k} Level_l(t),\ succ_k(n) = succ(n) \text{ if } n \in \bigcup_{l<k} Level_l(t), \text{ and } succ_k(n) = \emptyset \text{ if } n \in Level_k(t).$$

In this paper, we focus on finitely branching lts, and thus on finitary trees. Each finitary tree is univocally defined by the sequence of its projections: $\forall t, t' \in FyTree(\mathbb{A})((\forall k \in \mathbb{N} \pi_k(t) = \pi_k(t')) \Rightarrow t = t')$.

We consider domains of actions $(\mathbb{A}, \mathbf{d})$, where $\mathbf{d} : \mathbb{A} \times \mathbb{A} \to \mathbb{R}^+ \cup \{\infty\}$ is a distance between actions, with $\mathbf{d}(a,b) = \mathbf{d}(b,a)$, $\mathbf{d}(a,b) = 0 \Leftrightarrow a = b$, and $\mathbf{d}(a,c) + \mathbf{d}(c,b) \ge \mathbf{d}(a,b)$, $\forall a, b, c \in \mathbb{A}$, where $+$ is extended to $\mathbb{R}^+ \cup \{\infty\}$ as usual. We assume that $\mathbf{d}(a,b) = \infty$ when the value $\mathbf{d}(a,b)$ is not specified.

When comparing pairs of processes, it is natural [2, 3, 8] to introduce a *discount factor* $\alpha \in (0,1]$. Then, the differences in the k-th level of the compared trees are weighted by α^k, following the idea that differences in the far future are less important than those in the near. As a consequence, it is possible to obtain finite distances when comparing two processes with infinitely many differences.

In [15], we have presented, inter alia, an operational definition that allows us to obtain bounds for our global bisimulation distance between finite trees. These bounds are given by the cost of any transformation that turns one of the trees into the other. The following definition states which are the valid steps of those transformations and their costs. Roughly, any application of idempotency of $+$ has no cost, while the change of an action a at level k into another b, costs $\alpha^k \mathbf{d}(b,a)$. Intuitively, in what follows we write $t \rightsquigarrow^1_{\alpha,d} t'$ meaning that there is a distance step (aka 1-step transformation) between t and t' (with discount factor α) whose cost is at most d. We use the superscript 1 to distinguish between the 1-step relation $\rightsquigarrow^1_{\alpha,d}$ and its transitive closure $\rightsquigarrow_{\alpha,d}$.

Definition 4. *Given a domain of actions $(\mathbb{A}, \mathbf{d})$ and a discount factor $\alpha \in (0,1]$, we inductively define the distance steps on $FTrees(\mathbb{A})$ by*

1. $d \ge 0 \Rightarrow (t \rightsquigarrow^1_{\alpha,d} t + t \wedge t + t \rightsquigarrow^1_{\alpha,d} t)$.
2. $d \ge \mathbf{d}(a,b) \Rightarrow at \rightsquigarrow^1_{\alpha,d} bt$.
3. $t \rightsquigarrow^1_{\alpha,d} t' \Rightarrow t + t'' \rightsquigarrow^1_{\alpha,d} t' + t''$.
4. $t \rightsquigarrow^1_{\alpha,d} t' \Rightarrow at \rightsquigarrow^1_{\alpha,\alpha d} at'$.

We associate to each distance step its level. The level of any step generated by 1. or 2. is one; while if the level of the corresponding premise $t \rightsquigarrow^1_{\alpha,d} t'$ is k, then the level of a step generated by 3. (resp. 4.) is k (resp. $k+1$). Finally, we define the family of global distance relations $\langle \rightsquigarrow_{\alpha,d} | \ d \in \mathbb{R}^+ \rangle$, taking $t \rightsquigarrow_{\alpha,d} t'$ if there exists a sequence $\mathscr{S} := t = t^0 \rightsquigarrow^1_{\alpha,d_1} t^1 \rightsquigarrow^1_{\alpha,d_2} t^2 \rightsquigarrow^1_{\alpha,d_3} \cdots \rightsquigarrow^1_{\alpha,d_n} t^n = t'$, with $\sum_{i=1}^n d_i \le d$.

3 The coinductive global bisimulation distance

In order to extend our global distances to infinite trees, we have introduced in [17] a general coinductive notion of distance. We formalize our definition in two steps. In the first one we introduce the rules that

produce the steps of the *coinductive transformations* between trees, starting from any family of triples (t,t',d), with $t,t' \in FyTrees(\mathbb{A})$ and $d \in \mathbb{R}^+$.

Definition 5 ([17]). *Given a domain of actions $(\mathbb{A},\mathbf{d})$, a discount factor $\alpha \in (0,1]$ and a family $\mathscr{D} \subseteq FyTrees(\mathbb{A}) \times FyTrees(\mathbb{A}) \times \mathbb{R}^+$, we define the family of relations $\equiv_d^{\mathscr{D},\alpha}$, for $d \in \mathbb{R}^+$, by:*

1. *For all $d \geq 0$ we have (i) $(\sum_{j\in J} a_j t_j) + at + at \equiv_d^{\mathscr{D},\alpha} (\sum_{j\in J} a_j t_j) + at$,*
   and (ii) $(\sum_{j\in J} a_j t_j) + at \equiv_d^{\mathscr{D},\alpha} (\sum_{j\in J} a_j t_j) + at + at$.
2. *For all $d \geq \mathbf{d}(a,b)$ we have $(\sum_{j\in J} a_j t_j) + at \equiv_d^{\mathscr{D},\alpha} (\sum_{j\in J} a_j t_j) + bt$.*
3. *For all $(t,t',d) \in \mathscr{D}$ we have $(\sum_{j\in J} a_j t_j) + at \equiv_{d'}^{\mathscr{D},\alpha} (\sum_{j\in J} a_j t_j) + at'$ for all $d' \geq \alpha d$.*

To simplify the notation, we write $\equiv_d$ instead of $\equiv_d^{\mathscr{D},\alpha}$, whenever $\mathscr{D}$ and α are clear from the context.

We say that the steps generated by application of rules 1 and 2 in Def. 5 are *first level steps*; while those generated by rule 3 are *coinductive steps*. Inspired by the proof obligations imposed to bisimulations we introduce the coinductive proof obligations imposed to the (satisfactory) families of distances.

Definition 6 ([17]). *Given a domain of actions $(\mathbb{A},\mathbf{d})$ and a discount factor $\alpha \in (0,1]$, we say that a family $\mathscr{D}$ is an α-coinductive collection of distances (α-ccd) between finitary trees, if for all $(t,t',d) \in \mathscr{D}$ there exists a* finite coinductive transformation sequence *$\mathscr{C} := t = t^0 \equiv_{d_1} t^1 \equiv_{d_2} \ldots \equiv_{d_n} t^n = t'$, with $d \geq \sum_{j=1}^n d_j$. Then, when there exists an α-ccd $\mathscr{D}$ with $(t,t',d) \in \mathscr{D}$, we will write $t \equiv_d^\alpha t'$, and say that tree t is at most at distance d from tree t' wrt α.*

Example 1. *Let us consider the domain of actions $(\mathbb{N},\mathbf{d})$, where $\mathbf{d}$ is the usual distance for numbers, and the trees $t_N = unfold(N)$ and $t_{N'} = unfold(N')$, with $N = \{n_0,n_1\}$, $succ(n_0) = \{(0,n_0),(0,n_1)\}$ and $succ(n_1) = \emptyset$; and $N' = \{n'_0,n'_1\}$, $succ'(n'_0) = \{(0,n'_0),(1,n'_1)\}$ and $succ'(n'_1) = \emptyset$. Then, we have $t_N \equiv_2^{\mathscr{D},1/2} t_{N'}$, using the family $\mathscr{D} = \{(t_N,t_{N'},2)\}$. We can prove that this is indeed a $\frac{1}{2}$-ccd, by considering the sequence: $\mathscr{C} := t_N \equiv_1^{\mathscr{D},1/2} t_{N''} \equiv_1^{\mathscr{D},1/2} t_{N'}$, where $t_{N''} = unfold(N'')$, with $N'' = \{n''_0,n''_1,n''_2,n''_3\}$, $succ''(n''_0) = \{(1,n''_1),(0,n''_2)\}$, $succ''(n''_1) = \emptyset$, $succ''(n''_2) = \{(0,n''_2),(0,n''_3)\}$ and $succ''(n''_3) = \emptyset$.*

It is immediate to see that our notion of distance has natural and pleasant properties such as the *triangular transitivity*: for any discount factor $\alpha \in (0,1]$, whenever we have $t \equiv_d^\alpha t'$ and $t' \equiv_{d'}^\alpha t''$, we also have $t \equiv_{d+d'}^\alpha t''$. Of course, our coinductive definition of the global bisimulation distance generalizes our operational definition for finite trees.

Proposition 1 ([17]). *For $t,t' \in FTrees(\mathbb{A})$, the operational (Def. 4) and the coinductive (Def. 6) definition of global bisimulation distance between trees coincide, that means $t \equiv_d^\alpha t' \Leftrightarrow t \rightsquigarrow_{\alpha,d} t'$.*

Proof. $\underline{\Rightarrow}|$ We assume that $t \equiv_d^\alpha t'$ holds. This means that $(t,t',d) \in \mathscr{D}$ for some α-ccd $\mathscr{D}$, which in turn implies the existence of a finite coinductive sequence $\mathscr{C} := t = t^0 \equiv_{d_1} t^1 \equiv_{d_2} \ldots \equiv_{d_n} t^n = t'$, with $\sum d_j \leq d$. If $\mathscr{C}$ is the vacuous sequence ($n = 0$), then we have $t = t'$ and $t \rightsquigarrow_{\alpha,d} t'$ trivially holds. Let us assume $n > 0$. We show that it is possible to "unfold" $\mathscr{C}$ into a sequence of distance steps, $\mathscr{S}$, proving that $t \rightsquigarrow_{\alpha,d} t'$. It is enough to show that for a generic step $t^i \equiv_{d_{i+1}} t^{i+1}$ of $\mathscr{C}$ there exists a sequence proving that $t^i \rightsquigarrow_{\alpha,d_{i+1}} t^{i+1}$ (the complete sequence for $\mathscr{C}$ can be obtained by composition). If $t^i \equiv_{d_{i+1}} t^{i+1}$ has been obtained by applying rule 1 in Def. 5 (we only consider the sub-case (i), the other case can be dealt with in the same way), then we have $t^i = t^i_1 + at_1 + at_1 \equiv_{d_{i+1}} t^i_1 + at_1 = t^{i+1}$, and applying rules 1 and 3 in Def. 4 we get $t^i \rightsquigarrow_{\alpha,d_{i+1}} t^{i+1}$. If $t^i \equiv_{d_{i+1}} t^{i+1}$ has been obtained by applying rule 2 in Def. 5, then we have $t^i = t^i_1 + at_1 \equiv_{d_{i+1}} t^i_1 + bt_1 = t^{i+1}$, and applying rules 2 and 3 in Def. 4 we get $t^i \rightsquigarrow_{\alpha,d_{i+1}} t^{i+1}$. If $t^i \equiv_{d_{i+1}} t^{i+1}$ has been obtained by applying rule 3 in Def. 5, then we have $t^i = t^i_1 + at_1 \equiv_{d_{i+1}} t^i_1 + at'_1 = t^{i+1}$, with (t_1,t'_1,d') for some d' such that $d_{i+1} \geq \alpha d'$. We proceed by induction on the depth of t^i. Notice that $depth(t^i) > 0$ as there is no t'' such that $t^i \equiv_{d_{i+1}} t''$ when $depth(t^i) = 0$. If $depth(t^i) = 1$ (base case), then

$t_1 = \mathbf{0}$, and the only possible witness for (t_1, t_1', d') is the vacuous coinductive sequence. Thus, we have $t_1 = t_1' = \mathbf{0}$, and $t_1 \rightsquigarrow_{\alpha, d_{i+1}/\alpha} t_1'$ trivially holds. By applying rules 4 and 3 in Def. 4 we get $t^i \rightsquigarrow_{\alpha, d_{i+1}} t^{i+1}$. Finally, if $depth(t^i) > 1$, then $t_1 \rightsquigarrow_{\alpha, d'} t_1'$ holds by inductive hypothesis, and applying rules 4 and 3 in Def. 4 we get $t^i \rightsquigarrow_{\alpha, \alpha d'} t^{i+1}$, which in turn, trivially implies $t^i \rightsquigarrow_{\alpha, d_{i+1}} t^{i+1}$ (as $d_{i+1} \geq \alpha d'$).

$\underline{\Leftarrow}$ | Given a sequence of distance steps, $\mathscr{S}$, proving that $t \rightsquigarrow_{\alpha, d} t'$, we can "fold" it into a coinductive sequence, $\mathscr{C}$, witnessing $t \equiv_d^\alpha t'$. For each $(t, t', d) \in \mathscr{D}$ we consider the factorization of the sequence $\mathscr{S}$ and its reordering as done in [17]. We get $t = \sum_{i \in I_0} a_i t_i \rightsquigarrow_{\alpha, d_{02}} \sum_{i \in I_0} a_i t_i' \rightsquigarrow^1_{\alpha, d_{11}} \sum_{i \in I_1} a_i t_i \rightsquigarrow_{\alpha, d_{12}} \sum_{i \in I_1} a_i t_i' \rightsquigarrow^1_{\alpha, d_{21}} \cdots \rightsquigarrow_{\alpha, d_{(k+1)2}} \sum_{i \in I_{k+1}} a_i t_i' = t'$, where for each sequence $\sum_{i \in I_j} a_i t_i \rightsquigarrow_{\alpha, d_{j2}} \sum_{i \in I_j} a_i t_i'$ and each $i \in I_j$, we have $t_i \rightsquigarrow_{\alpha, d^i_{j2}/\alpha} t_i'$ with $\sum_{i \in I_j} d^i_{j2} = d_{j2}$. Now, applying the induction hypothesis, we have $(t_i, t_i', d^i_{j2}/\alpha) \in \mathscr{D}$, for all $i \in I_j$, so that $\sum a_i t_i \equiv^{\mathscr{D}, \alpha}_{\alpha d_1} \sum a_i t_i^1 \equiv^{\mathscr{D}, \alpha}_{\alpha d_2} \sum a_i t_i^2 \equiv^{\mathscr{D}, \alpha}_{\alpha d_3} \cdots \equiv^{\mathscr{D}, \alpha}_{\alpha d_{|I_j|}} \sum a_i t_i^{|I_j|} = \sum a_i t_i'$.

Therefore, each sequence $\sum_{i \in I_j} a_i t_i \rightsquigarrow_{\alpha, d^i_{j2}} \sum_{i \in I_j} a_i t_i'$ at the factorization above can be substituted by a sequence of $|I_j|$ valid coinductive steps, getting a total distance $\sum_{j=0}^{k+1} \sum_{i \in I_j} d^i_{j1} + \sum_{j=0}^{k+1} \sum_{k=1}^{|I_j|} \alpha d_k = d$. □

Even if the result above only concerns finite trees, it reveals the duality between induction and coinduction. Of course, its consequences are much more interesting in the infinite case.

Example 2. *Let $(\{a, b\}, \mathbf{d})$ be a domain of actions such that $\mathbf{d}(a, b) = 1$ and let us consider the lts given by $N_{1,\infty} = \{n_0\}$, with $succ(n_0) = \{(a, n_0)\}$ and its unfolding a^∞. In an analogous way, we obtain the tree b^∞. We have $\pi_n(a^\infty) = a^n$, and clearly a^∞ can be seen as the "limit" of its projections. Applying Def. 4, we obtain $a^n \rightsquigarrow_{1/2,2} b^n$ and thus $a^n \equiv_2^{1/2} b^n$. We also have $a^\infty \equiv_2^{1/2} b^\infty$, which can indeed be proved by means of the (trivial!) collection $\mathscr{D} = \{(a^\infty, b^\infty, 2)\}$. We can check that $\mathscr{D}$ is an $\frac{1}{2}$-ccd using the coinductive sequence $\mathscr{C} := a^\infty = aa^\infty \equiv_1^{\mathscr{D}, 1/2} ba^\infty \equiv^{\mathscr{D}, 1/2}_{\frac{1}{2} \cdot 2} bb^\infty = b^\infty$.*

4 On the continuity of the global bisimulation distance

It is very simple to prove the following proposition, by introducing the notion of projections of α-ccd's.

Proposition 2 ([17])**.** *For any α-ccd $\mathscr{D}$, the projected family $\pi(\mathscr{D}) = \{(\pi_n(t), \pi_n(t'), d) \mid (t, t', d) \in \mathscr{D}, n \in \mathbb{N}\}$ is an α-ccd. Hence, it holds $t \equiv_d^\alpha t' \Rightarrow \pi_n(t) \equiv_d^\alpha \pi_n(t')$ for each $n \in \mathbb{N}$.*

Proof. Let $\mathscr{C} := t = t^0 \equiv^{\mathscr{D}, \alpha}_{d_1} \ldots \equiv^{\mathscr{D}, \alpha}_{d_k} t^k = t'$ be the coinductive sequence proving that $(t, t', d) \in \mathscr{D}$ satisfies the condition in order $\mathscr{D}$ to be an α-ccd. Then each projected sequence $\pi_n(\mathscr{C}) := \pi_n(t) = \pi_n(t^0) \equiv^{\pi(\mathscr{D}), \alpha}_{d_1} \ldots \equiv^{\pi(\mathscr{D}), \alpha}_{d_k} \pi_n(t^k) = \pi_n(t')$ proves that $(\pi_n(t), \pi_n(t'), d) \in \pi(\mathscr{D})$ satisfies the condition in order $\pi(\mathscr{D})$ to be an α-ccd. It is clear that the projection under π_n of any first level step in $\mathscr{C}$, is also a valid step in $\pi_n(\mathscr{C})$. Moreover, any coinductive step in $\mathscr{C}$ using $(t_1, t_1', d') \in \mathscr{D}$, can be substituted by the corresponding projected step, that uses $(\pi_{n-1}(t_1), \pi_{n-1}(t_1'), d') \in \pi(\mathscr{D})$. □

Remark 1. *Alternatively, we can consider for each $n \in \mathbb{N}$ a family $\mathscr{D}_n = \pi_n(\mathscr{D}) = \{(\pi_m(t), \pi_m(t'), d) \mid (t, t', d) \in \mathscr{D}, m \in \mathbb{N}, m \leq n\}$, using the fact that the subtrees of a projection $\pi_n(t)$ are also projections $\pi_m(t'')$ of subtrees t'' of t, for some $m < n$. These families satisfy $\pi_m(\mathscr{D}) \subseteq \pi_n(\mathscr{D})$, whenever $m \leq n$.*

We conjecture that the converse of Prop. 2, asserting the continuity of our coinductive distance, is also valid. However, after several attempts have failed, or led us into a blind alley, we are now less confident of this result than we were before. We expected to be able to prove it by using the argument used in the proof of Prop. 2, in the opposite direction. If we would have a collection of "uniform"[1]

[1] Uniformity here means that for any $n, k \in \mathbb{N}$, with $k \geq n$, the steps corresponding to the first n levels of any sequence $\mathscr{S}^k$ are always the same.

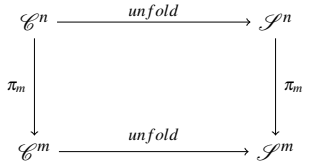


Figure 1: Relating projections of sequences

operational sequences $\mathscr{S}^n := \pi_n(t) \rightsquigarrow_{\alpha,d} \pi_n(t')$, we could "overlap" all of them getting an "infinite tree-structured sequence". By "folding" it we would obtain the coinductive sequence proving that $t \equiv_d^\alpha t'$. Next, a simple example.

Example 3 ([17])**.** *Let us consider the trees $t = ac^\infty + ad^\infty$ and $t' = bc^\infty + bd^\infty$, and the distance* **d** *defined by* $\mathbf{d}(a,b) = 4$, $\mathbf{d}(c,d) = 1$. *Then we have (the numbers above the arrows denote the distance step level, according to Def. 4; moreover, even if the arrows denote here 1-step transformations, we omit the superscript 1 for the sake of readability):*

$$\pi_1(t) = a + a \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},0} a \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},4} b \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},0} b + b = \pi_1(t'),$$
$$\pi_2(t) = ac + ad \overset{(2)}{\rightsquigarrow}_{\frac{1}{2},\frac{1}{2}\cdot 1} ac + ac \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},0} ac \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},4} bc \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},0} bc + bc \overset{(2)}{\rightsquigarrow}_{\frac{1}{2},\frac{1}{2}\cdot 1} bc + bd = \pi_2(t'),$$
$$\pi_3(t) = acc + add \overset{(2)}{\rightsquigarrow}_{\frac{1}{2},\frac{1}{2}\cdot 1} acc + acd \overset{(3)}{\rightsquigarrow}_{\frac{1}{2},\frac{1}{4}\cdot 1} acc + acc \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},0} acc \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},4}$$
$$bcc \overset{(1)}{\rightsquigarrow}_{\frac{1}{2},0} bcc + bcc \overset{(2)}{\rightsquigarrow}_{\frac{1}{2},\frac{1}{2}\cdot 1} bcc + bdc \overset{(3)}{\rightsquigarrow}_{\frac{1}{2},\frac{1}{4}\cdot 1} bcc + bdd = \pi_3(t').$$

Now, if we consider the operational sequences, $\mathscr{S}^n$, relating $\pi_n(t)$ and $\pi_n(t')$, for any $n \in \mathbb{N}$, we obtain $\pi_n(t) \rightsquigarrow_{\frac{1}{2},d_n} \pi_n(t')$, for some $d_n < 6$. Applying Prop. 1 we can turn these operational sequences into equivalent coinductive ones, $\mathscr{C}^n$, thus proving $\pi_n(t) \equiv_6^{1/2} \pi_n(t')$, for all $n \in \mathbb{N}$:

$$\mathscr{C}^1 := a + a \equiv_0 a \equiv_4 b \equiv_0 b + b,$$
$$\mathscr{C}^2 := ac + ad \equiv_{\frac{1}{2}} ac + ac \equiv_0 ac \equiv_4 bc \equiv_0 bc + bc \equiv_{\frac{1}{2}} bc + bd,$$
$$\mathscr{C}^3 := acc + add \equiv_{\frac{1}{2}} acc + acd \equiv_{\frac{1}{4}} acc + acc \equiv_0 acc \equiv_4 bcc \equiv_0 bcc + bcc \equiv_{\frac{1}{2}} bcc + bdc \equiv_{\frac{1}{4}} bcc + bdd.$$

Lemma 1. *If $t \rightsquigarrow^1_{\alpha,d} t'$ is a distance step of level l, then $\pi_k(t) \rightsquigarrow^1_{\alpha,d} \pi_k(t')$ is also a distance step of level l for all $k \geq l$. Instead, if $l > k$ then we have $\pi_k(t) = \pi_k(t')$.*

The continuity of our coinductive distance would mean that, whenever we have $\pi_n(t) \equiv_d^\alpha \pi_n(t')$ $\forall n \in \mathbb{N}$, there should be a collection of "uniform" sequences proving these facts. Certainly, this is the case when we know in advance that $t \equiv_d^\alpha t'$. Next, we define the projection of our operational sequences.

Definition 7. *Given a sequence of distance steps $\mathscr{S} := t = t^0 \rightsquigarrow^1_{\alpha,d_1} t^1 \rightsquigarrow^1_{\alpha,d_2} \cdots \rightsquigarrow^1_{\alpha,d_n} t^n = t'$ with $d = \sum_{i=1}^n d_i$, for any $k \in \mathbb{N}$ we define the sequence $\pi_k(\mathscr{S}) := \pi_k(t) = \pi_k(t^0) \rightsquigarrow^1_{\alpha,d_{i_1}} \pi_k(t^{i_1}) \rightsquigarrow^1_{\alpha,d_{i_2}} \cdots \rightsquigarrow^1_{\alpha,d_{i_l}}$ $\pi_k(t^{i_l}) = \pi_k(t')$, where $\langle i_1, i_2, \ldots, i_l \rangle$ is the sequence of indexes i_j for which the level of the distance step $t^{i_j-1} \rightsquigarrow^1_{\alpha,d_{i_j}} t^{i_j}$ is less than or equal to k, while the rest of the steps are removed.*

Proposition 3. *For any $k \in \mathbb{N}$, and any sequence of distance steps $\mathscr{S}$, the projected sequence $\pi_k(\mathscr{S})$ is a sequence of distance steps, thus proving $\pi_k(t) \rightsquigarrow_{\alpha,\sum d_{i_j}} \pi_k(t')$. Therefore, we also have $\pi_k(t) \rightsquigarrow_{\alpha,d} \pi_k(t')$.*

Now, let us start with a coinductive sequence $\mathscr{C}$ relating two (possibly finitary) trees t and t'.

Corollary 1. *The projection of coinductive sequences and those of distance steps that the former induces, are related by the commutative diagram in Fig. 1. There we denote by $\mathscr{C}^n$ the projections of a coinductive sequence $\mathscr{C}$.*

Note that all these coinductive sequences have exactly the "same structure", which is formalized by the fact that $\pi_m(\mathscr{C}^n) = \mathscr{C}^m$, for all $m \leq n$. As a consequence, if now we forget that we have already the original family $\mathscr{D}$ in Ex. 2, starting from the families $\mathscr{D}_n$ in Remark 1, we could "reverse" the procedure by means of which they were defined, obtaining a single family $\mathscr{D}' = \bigcup \mathscr{D}_n \cup \{(a^\infty, b^\infty, 2)\} = \{(a^k, b^k, 2) \mid k \leq \infty\}$, where we have added the "limit" triple $(a^\infty, b^\infty, 2)$ because for all $k \in \mathbb{N}$ $(\pi_k(a^\infty), \pi_k(b^\infty), 2) = (a^k, b^k, 2) \in \bigcup \mathscr{D}_n$. Now, we can see that $\mathscr{D}'$ is indeed a $\frac{1}{2}$-ccd. In order to check the condition corresponding to $(a^\infty, b^\infty, 2)$, we "overlap" the sequences $\mathscr{C}^n$ getting a sequence $\mathscr{C}'$ constructed as follows: From the first level step $a^n = aa^{n-1} \equiv_1^{\mathscr{D}_n,1/2} ba^{n-1}$ at each $\mathscr{C}^n$, we obtain the first level step $a^\infty = aa^\infty \equiv_1^{\mathscr{D}',1/2} ba^\infty$; this is followed by the coinductive step $ba^\infty \equiv_{\frac{1}{2}\cdot 2}^{\mathscr{D}',1/2} bb^\infty = b^\infty$, obtained just removing the projections from any of the coinductive steps $ba^{n-1} = \pi_n(ba^\infty) \equiv_{\frac{1}{2}\cdot 2}^{\mathscr{D}_n,1/2} \pi_n(bb^\infty) = \pi_n(b^\infty) = b^n$.

The important fact about the construction above is that it can be applied to any collection of coinductive sequences that "match each other". We will call these collections *telescopic*, because when we "unfold" their elements we obtain a sequence of operational sequences, where any of them is obtained from the previous one by adding some new intermediate steps, that always correspond to transformation steps at the last level of the compared trees (this reminds us the "opening" of a telescopic antenna).

Definition 8. *Let $t, t' \in FyTrees(\mathbb{A})$ and let $(\mathscr{S}^n)_{n\in\mathbb{N}}$ be a collection of operational sequences proving $\pi_n(t) \rightsquigarrow_{\alpha,d} \pi_n(t')$. We say that it is* telescopic[2] *if $\pi_m(\mathscr{S}^n) = \mathscr{S}^m$, for all $m, n \in \mathbb{N}$, with $m \leq n$.*

Any telescopic collection $(\mathscr{S}^n)_{n\in\mathbb{N}}$, produces a "limit" coinductive sequence $\mathscr{C}$ proving $t \equiv_d^\alpha t'$:

Lemma 2. *Let $(\mathscr{S}^n)_{n\in\mathbb{N}}$ be a telescopic collection relating t and t'. We consider the associated factorization of each sequence in it: $\mathscr{S}^n := \pi_n(t) = t^{n,0,1} \rightsquigarrow_{\alpha,d_{n02}} t^{n,0,2} \rightsquigarrow_{\alpha,d_{n11}}^1 t^{n,1,1} \dots \rightsquigarrow_{\alpha,d_{nk1}}^1 t^{n,k,1} \rightsquigarrow_{\alpha,d_{nk2}} t^{n,k,2} = \pi_n(t')$ where we alternate distance steps at the first level and global steps, that aggregate a sequence of steps at deeper levels. Then, for all $m \leq n$, $j \in \{1,\dots,k\}$ and $r \in \{1,2\}$, we have $t^{m,j,r} = \pi_m(t^{n,j,r})$.*

Proof. Immediate, by definition of projections and telescopic collections. □

Corollary 2. *For all $n \in \mathbb{N}$, $j \in \{1,\dots,k\}$ and $r \in \{1,2\}$, we can decompose $t^{n,j,r}$ as above into $\sum_{i=1}^{I_{njr}} a_i t_i^{n,j,r}$, where the sequences $t^{n,j,1} \rightsquigarrow_{\alpha,d_{nj2}} t^{n,j,2}$ satisfy $I_{nj2} = I_{nj1}$ $\forall j \in \{1\dots k\}$, and can be factorized into: $t^{n,j,1} = \sum a_i t_i^{n,j,1} \rightsquigarrow_{\alpha,d_{nj2,1}} \sum a_i t_i^{n,j,21} \cdots \rightsquigarrow_{\alpha,d_{nj2,I_{nj2}}} \sum a_i t_i^{n,j,2I_{nj2}} = \sum a_i t_i^{n,j,2} = t^{n,j,2}$, where $t_i^{n,j,2l} = t_i^{n,j,2}$ $\forall i \leq l$ and $t_i^{n,j,2,l} = t_i^{n,j,1}$ $\forall i > l$, and $\sum_{i=1}^{I_{nj2}} d_{nj2,i} = d_{nj2}$.*

As a consequence, if we denote by $t^{\infty,j,r}$ the unique tree in $FyTree$ that satisfies $\pi_n(t^{\infty,j,r}) = t^{n,j,r}$ $\forall n \in \mathbb{N}$, we can decompose it into $\sum_{i=1}^{I_{j,r}} a_i t_i^{\infty,j,r}$, and each collection $(\mathscr{S}_{j,i}^n)_{n\in\mathbb{N}}$ given by $\mathscr{S}_{j,i}^n := \pi_n(t_i^{\infty,j,1}) = t_i^{n,j,1} \rightsquigarrow_{\alpha,d_{nj2}} t_i^{n,j,2} = \pi_n(t_i^{\infty,j,2})$ is indeed a telescopic collection relating $t_i^{\infty,j,1}$ and $t_i^{\infty,j,2}$.

Proof. Easy to check, by definition of projections and telescopic collections. □

Theorem 1. *Let $\mathscr{D} = \{(t, t', d) \mid \exists\, (\mathscr{S}^n)_{n\in\mathbb{N}}$ a telescopic collection relating t and $t'\}$, then $\mathscr{D}$ is an α-ccd.*

Proof. Using the notation in Lem. 2 and Cor. 2 above, we can consider the coinductive sequence

$$t = t^{\infty,0,1} \equiv_{d_{02}}^{\mathscr{D},\alpha} t^{\infty,0,2} \equiv_{d_{11}}^{\mathscr{D},\alpha} t^{\infty,1,1} \equiv_{d_{12}}^{\mathscr{D},\alpha} \cdots \equiv_{d_{k2}}^{\mathscr{D},\alpha} t^{\infty,k,2} = t',$$

[2] Certainly, these "telescopic" sequences correspond to the notion of inverse limit in domain theory or category theory. But since we only need a very concrete case of that quite abstract notion, we prefer to define it in an explicit way here.

where $d_{j2} = sup_{n\in\mathbb{N}}\{d_{nj1}\}$ and $d_{j1} = d_{nj1}, \forall n \in \mathbb{N}$. It is clear that all the steps $t^{\infty,j,2} \equiv^{\mathscr{D},\alpha}_{d_{j1}} t^{\infty,j+1,1}$ are valid first level steps, while using Cor. 2 we have that the steps $t^{\infty,j,1} \equiv^{\mathscr{D},\alpha}_{d_{j2}} t^{\infty,j,2}$ correspond to valid coinductive steps from $\mathscr{D}$. This is so because joining all the members of the telescopic sequences $(\mathscr{S}^n_{j,i})_{n\in\mathbb{N}}$ with $j \in \{0\ldots k\}$ fixed, we obtain a single telescopic sequence $(\mathscr{S}^n_j)_{n\in\mathbb{N}}$ relating $t^{\infty,j,1} = \sum_{i=1}^{I_{j,r}} a_i t_i^{\infty,j,1}$ and $t^{\infty,j,2} = \sum_{i=1}^{I_{j,r}} a_i t_i^{\infty,j,2}$. □

If for any $t,t' \in FyTrees$ there would be only finitely many sequences proving each valid triple $t \rightsquigarrow_{\alpha,d} t'$, then a classical compactness technique (or König's lemma, if you prefer) would immediately prove that whenever we have $\pi_n(t) \equiv^{\alpha}_{d} \pi_n(t')$ for any $n \in \mathbb{N}$, we can obtain a telescopic collection of sequences proving all these facts. Then, the application of Th. 1 would conclude the continuity of our global bisimulation distance. Unfortunately, this is not the case, because we can arbitrary enlarge any such sequence, adding dummy steps that apply the idempotency rule in one and the other directions. Even more, in some complicated cases in order to obtain some distances between two trees we need to consider sequences that include intermediate trees wider than the compared ones.

Example 4. *Let $\mathbb{A} = \{1,2,3,4,5\}$ with the "usual" distance $\mathbf{d}(n,m) = |m-n|$, for all $m,n \in \mathbb{A}$. Let us consider the trees $t = 1(2+3+4+5)+1(1+2+3+4)$ and $t' = 1(1+2+4+5)$. Then, we have $t \rightsquigarrow_{1,3} t'$, which can be obtained by means of the sequence $\mathscr{S} := t \rightsquigarrow^1_{1,0} 1(2+2+3+4+5)+1(1+2+3+4) \rightsquigarrow^1_{1,1} 1(1+2+3+4+5)+1(1+2+3+4) \rightsquigarrow^1_{1,0} 1(1+2+3+4+5)+1(1+2+3+4+4) \rightsquigarrow^1_{1,1} 1(1+2+3+4+5)+1(1+2+3+4+5) \rightsquigarrow^1_{1,0} 1(1+2+3+4+5) \rightsquigarrow^1_{1,1} 1(1+2+4+4+5) \rightsquigarrow^1_{1,0} 1(1+2+4+5) = t'$.*

Proposition 4. *There exist $t,t' \in FTrees(\mathbb{A})$, with $t \rightsquigarrow_{\alpha,d} t'$ and $||t||_m$, $||t'||_m \leq k$ for some $m,k \in \mathbb{N}$, for which it is not possible to obtain an operational sequence $\mathscr{S}$ witnessing $t \rightsquigarrow_{\alpha,d} t'$ whose intermediate trees t^i satisfy $||t^i||_m \leq k$.*

Proof. For the trees t, t' in Ex. 4, there is no sequence $\mathscr{S}$ proving $t \rightsquigarrow^1_{1,3} t'$ that only includes intermediate trees t'' with $||t''||_2 \leq 4$.

We have to transform the sets $A = \{2,3,4,5\}$ and $B = \{1,2,3,4\}$ into $C = \{1,2,4,5\}$. First of all, we need to unify A and B, otherwise if we try to get C from A and B independently it will cost more than 3 units. In order to unify A and B, we need at least a 2 units payment getting one of the following intermediate sets: $C'_1 = \{1,2,3,4,5\}$, $C'_2 = \{1,2,3,4\}$, $C'_3 = \{2,3,4,5\}$ or $C'_4 = \{2,3,4\}$. Now, we see that C'_1 is the only intermediate set that gives us the 3 units cost given in Ex. 4.

$C'_2 \rightsquigarrow_{1,2} C$: Pay 1 unit to add 5, and another to "erase" 3, so that the total cost would be 4.

$C'_3 \rightsquigarrow_{1,2} C$: Pay 1 unit to add 1, and another to "erase" 3, so that the total cost would be 4 again.

$C'_4 \rightsquigarrow_{1,3} C$: Pay 1 unit to add 1, another to add 5, and another to "erase" 3; the total cost would be 5. □

Also, the following example shows us that in some cases the sequences at the telescopic sequences could be a bit involved. Even if for small values of m the projections $\pi_m(t)$ and $\pi_m(t')$ could be "quite close", we need more elaborated sequences for relating them. Otherwise we would not be able to expand those sequences into others connecting $\pi_n(t)$ and $\pi_n(t')$, for larger values of n.

Example 5. *We can check $ac+bd \rightsquigarrow_{1,2} ad+bc$ by using the sequence $\mathscr{S} := ac+bd \rightsquigarrow^1_{1,1} bc+bd \rightsquigarrow^1_{1,1} bc+ad = ad+bc$, which corresponds to $\pi_1(\mathscr{S}) = \mathscr{S}^1 := a+b \rightsquigarrow^1_{1,1} b+b \rightsquigarrow^1_{1,1} b+a = a+b$. Therefore, in this case, we cannot start with the trivial sequence $\mathscr{S}'^1 := a+b = b+a$, and moreover we need to take the order of "summands" into account, expressing the fact that we have really to change a into b, and b into a.*

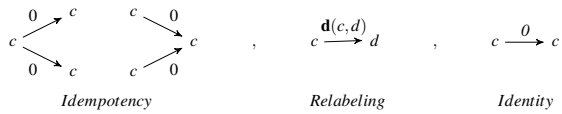


Figure 2: Arcs of the graph representing the transformation of single level trees.

5 Some partial results looking for continuity

Ex. 4 shows us that sometimes we need to use intermediate trees containing subtrees that are wider than the corresponding ones in the two compared trees. However, we expected to prove that the width of those subtrees would be bounded by some (simple) function of the width of the subtrees at the same depth of the compared trees. We tried a proof by induction on the depth of the trees, whose base case should correspond to depth 1.

We consider 1-depth trees $t = \sum_{i=1}^{n} a_i$ and $t' = \sum_{j=1}^{m} b_j$, and the corresponding multiset of labels $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_m\}$. Now any step in a sequence $t \rightsquigarrow_{\alpha,d} t'$ is, of course, a first level step and corresponds to the application of either idempotency, in any direction, or the relabeling rule. We can represent these sequences by means of a multi-stage graph. We formally denote these graphs by (S, T, l, v), where (i) S is partitioned into $\{S_j\}_{j=0,\ldots,m}$, (ii) $l\,|_{S_0}\colon S_0 \to A$ and $l\,|_{S_m}\colon S_m \to B$ are bijections, and (iii) arcs of T are of the form (t,u), with $t \in S_i$ and $u \in S_{i+1}$ for some i, and correspond to any of the patterns in Fig. 2, with the application of a single non-identity pattern at each stage. Then, the full cost d of the sequence is just the sum of the costs of all the arcs in the graph.

Definition 9. *We say that a multi-stage graph is*

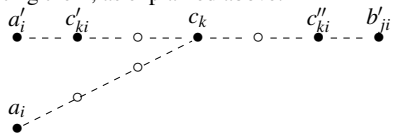
1. Totally sides connecting *(tsc) if for all $s_0 \in S_0$ there exists a path connecting it with some $s_m \in S_m$, and, symmetrically, for all $s_m \in S_m$ there exists a path connecting it with some $s_0 \in S_0$.*
2. Totally both ways connected *(tbwc) if it is tsc and for each $j \in \{1, \ldots, m-1\}$ and each $s \in S_j$ there exist two arcs $(s', s), (s, s'') \in T$.*

Proposition 5. *Let $t = \sum_{i=1}^{n} a_i$ and $t' = \sum_{j=1}^{m} b_j$ as above. Then, whenever we have $t \rightsquigarrow_{\alpha,d} t'$, we can prove this by means of a sequence that has at most $3(m+n-2)+1$ distance steps, and thus the width of the trees along it are also bounded by that amount.*

Proof. We consider the multi-stage graph $\mathscr{G}$ that represents the sequence $t \rightsquigarrow_{\alpha,d} t'$. We observe that this graph is tsc, using this fact we obtain a "compacted" subgraph $\mathscr{H}$ (of the original graph $\mathscr{G}$) by selecting a subset of the nodes of $\mathscr{G}$ and disjoints paths connecting them. So that, ever node in $\mathscr{G}$-$\mathscr{H}$ is at most in one of theses paths.

We turn these paths into arcs of $\mathscr{H}$ whose cost is $d(c_i, c_j)$, where c_i, c_j are the two extremes of the path. By applying the triangle inequality, we immediately obtain that this cost is no larger than the cost of the original path. Therefore, the full cost of $\mathscr{H}$ is no bigger than that of $\mathscr{G}$. The set of nodes in $\mathscr{H}$ is obtained in the following way:

- First, we consider $a_1 \in A$ and some reachable $b_{j1} \in B$ from it. We introduce both of them in $\mathscr{H}$, and also add the arc connecting them, as explained above.



- Next, we consider each of the remaining $a_i \in A$ and we select again some b_{ji} with which it is connected. We take the path in $\mathscr{G}$ connecting them, and if it does not cross any path in $\mathscr{G}$ that generated an arc in $\mathscr{H}$, then we proceed as in the first case. Otherwise, we consider the first arc $\langle c'_{ki}, c''_{ki}\rangle$ in $\mathscr{H}$ "traversed" by the new path. If c_{ki} is the common node to the two involved paths, then we add it to $\mathscr{H}$ and remove the arc $\langle c'_{ki}, c''_{ki}\rangle$ adding instead two new arcs $\langle c'_{ki}, c_{ki}\rangle$ and $\langle c_{ki}, c''_{ki}\rangle$, together with the arc $\langle a_i, c_{ki}\rangle$.

Finally, we proceed in the same way, but going backwards in the graph, for every $b_j \in B$ that was not still reached from any $a_i \in A$. Clearly, at any stage of the construction we add two new arcs in the worst case, and besides we need an idempotency step each time we consider a path that crosses $\mathscr{H}$. This finally produces a sequence from t to t' with at most $3(m+n-2)+1$ steps, whose cost is not bigger than that of the original sequence. Since the width of the trees change at most one at any step, the results about the bound of these widths follows immediately. □

The important thing about the bounds obtained above, is that they only depend on the cardinality of the multisets A and B, but no at all on the properties of the domain of actions $(\mathbb{A}, \mathbf{d})$. Even more, we can extend this result to any two finite trees $t = \sum_{a\in A} at_a$ and $t' = \sum_{b\in B} bt_b$: the size and complexity of the "continuations" t_a and t_b do not compromise the bound on the number of first level steps of a sequence relating t and t', that bound only depends on $||t||_1$ and $||t'||_1$.

Proposition 6. *Let $t = \sum_{i=1}^{n} a_i t_i$ and $t' = \sum_{j=1}^{m} b_j t_j$ be two trees such that $t \equiv_d^\alpha t'$. Then, we can prove this by means of a coinductive transformation sequence $\mathscr{C}$, that has at most $3(m+n-2)+1$ first level steps.*

Proof. We observe that the result in Prop. 5 could be obtained in exactly the same way if instead of a distance function on A, we would consider a relation $d \subseteq A \times A \times \mathbb{R}^+$ that satisfies the properties that define "bounds for a distance" in A:

- $\forall a \in A\ d(a,a,d)\ \forall d \in \mathbb{R}^+$.
- $d(a,b,d) \Leftrightarrow d(b,a,d)$.
- $d(a,b,d_1) \wedge d(b,c,d_2) \Rightarrow d(a,c,d_1+d_2)$.

Then, we consider the set of "prefixed" trees $\mathbb{A}FTrees = \mathbb{A} \times FTrees$, and we take $d_\alpha(at, bt', d_1+d_2)$ $\Leftrightarrow d_1 = \mathbf{d}(a,b) \wedge t \rightsquigarrow_{\alpha, d_2/\alpha} t'$. It is clear that for any distance d on A, and any $\alpha \in \mathbb{R}^+$, each such d_α defines bounds for a distance in A. Now, we can see each finite trees $t = \sum_{i=1}^{n} a_i t_i$ as a 1-depth trees $t = \sum_{i=1}^{n} \langle a_i, t_i\rangle$, for the alphabet $\mathbb{A}FTrees$. For any $\alpha \in \mathbb{R}^+$ we can consider the "bound for a distance" relation d_α, and apply Prop. 5 to conclude the proof. □

5.1 An alternative proof of the bounds for the first level

Even if the bounds obtained in Prop. 5 are rather satisfactory we have been able to prove some tighter bounds by reducing the induced graph by means of local simplifications rules. We consider interesting to show this proof because they bring to light how we could proceed when transforming the trees all along the sequences (even if we have not been able to successfully use this ideas in order to fully prove the desired continuity result).

Proposition 7. *Any multi-stage graph that is tsc can be turned into a tbwc multi-stage graph that is a subgraph of the original one.*

Proof. We repeatedly remove any intermediate node which is not two ways connected, and it is clear that after any such removal the graph remains tsc. □

Proposition 8. *1. The multi-stage graph associated to a sequence proving that $A \equiv_d^\alpha B$ is tbwc.*

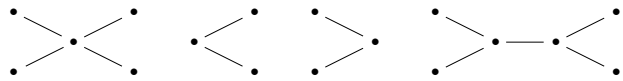


Figure 3: Schematic examples of diabolos

2. *Any subgraph of such a multi-stage graph, obtained by the removal of some internal nodes, that still remains tbwc, can also be obtained from some sequence proving $A \equiv_d^\alpha B$, which will be shorter or have the same length than the initial sequence.*

Proof. (1) Trivial. (2) When removing an internal node we are possibly removing an application of idempotency that turns into a simple identity arc. In such a case, we can remove this stage of the sequence getting a shorter sequence still proving $A \equiv_d^\alpha B$. □

Remark 2. *The result in Prop. 8.2 is true, not only if we remove nodes that are not both ways connected (in fact, there is no such in a tbwc multi-stage graph!), but also if we remove any subset of intermediate nodes, as far as the graph remains tsc. This is what we next use in order to reduce the size of the graph.*

Definition 10. *We say that a tbwc multi-stage graph is a* diabolo *if there exists some stage $i \in \{0 \dots m\}$ with $|S_i| = 1$, such that: (i) for all $j < i$ and all $s_j \in S_j$, $|\{s' \mid (s_j, s') \in T\}| = 1$, and (ii) for all $j > i$ and all $s_j \in S_j$, $|\{s' \mid (s', s_j) \in T\}| = 1$. We say that $s_i \in S_i$ is a center of the diabolo.*

Next we prove that we can reduce any tbwc multi-stage graph into a disjoint union of diabolos. We will do it by removing some "redundant" arcs, in such a way that these removals do not destroy the *tsc* character of the graph.

Definition 11. 1. *We say that a sequence of consecutive arcs*

$$(s_j, s_{j+1}), (s_{j+1}, s_{j+2}) \dots (s_{j+k-1}, s_{j+k})$$

in a tsc graph is a join sequence *if: (i) for all intermediate nodes in the sequence s_{j+l} with $l \in \{1, \dots, k-1\}$, the two arcs in the sequence, (s_{j+l-1}, s_{j+l}) and (s_{j+l}, s_{j+l+1}) are the only ones in T that involve s_{j+l}; and (ii) there are at least two other arcs (s_j, s') and (s'', s_{j+l}) in T that are not in the sequence.*

2. *We say that a node $s_j \in S_j$ is left-reducible (resp. right-reducible) if it can only be reached from a single node $s_0 \in S_0$ (resp. $s_m \in S_m$), but there are several arcs $(s'_{j-1}, s_j) \in T$ (resp. $(s_j, s'_{j+1}) \in T$).*

Join Sequence *Left-reducible Node*

Proposition 9. 1. *Whenever we have a join sequence in a tbwc graph, we can remove all its arcs and the intermediate nodes, preserving the tbwc property.*

2. *Whenever we have a left (resp. right)-reducible node s_j in a tbwc, we can remove one of the arcs (s'_{j-1}, s_j) reaching (resp. (s_j, s'_{j+1}) leaving) the node, preserving the tsc property.*

Proof. (1) Clearly, after the removal the graph remains tbwc, due to the existence of the two "lateral" arcs (s_j, s'_{j+1}) and (s'_{j+l-1}, s_{j+l}). (2) It is clear that after the removal we have still another arc reaching (resp. leaving) the reducible node from the same side. This can be still used to reach the corresponding node $s_0 \in S_0$ (resp. $s_m \in S_m$). And no other node in either S_0 and S_m is affected by the removal. □

Theorem 2. *By removing some intermediate nodes we can reduce any tbwc multi-stage graph into another such graph (with smaller or equal total cost) which is the disjoint union of a collection of diabolos.*

Proof. We start by reducing the graph applying Prop. 9, until we cannot apply it anymore. We obtain a tsc graph connecting the same sets of nodes S_0 and S_m, which can be turned into another tbwc (smaller) by applying Prop.7. By abuse of notation, let us still denote by $S_1, \ldots, S_{m-1}$ the other stages of the graph.

- First, we consider the connected components of the graph containing exactly one node s_0 belonging to S_0. These components are right-degenerated diabolos with s_0 as center. Indeed, if this was not the case, then the component would still contain a left-reducible node.
- Now, let us consider a connected component containing a subset of nodes $\{s_0^1, \ldots, s_0^k\} \subseteq S_0$, with $k > 1$. It is not difficult to verify that for each s_0^i ($1 \leq i \leq k$), there is exactly one arc leaving s_0^i: otherwise, the connected component would still contain a join sequence (due to the presence of multiple nodes in S_0). Using a similar argument, it is possible to prove that there can be only a single arc leaving each node in the following stages as well, until a stage i is reached such that the component contains exactly one node $s_i \in S_i$. Reasoning in a symmetric way from the right side (set S_m), it is eventually possible to show that the considered component is a diabolo.

Therefore, after the reduction, the graph is indeed the union of a family of disjoint diabolos. □

Proposition 10. *1. A diabolo connecting two sets of nodes S_0 and S_m satisfies $|S_i| \leq max\{|S_0|, |S_m|\}$ for each i.*

2. Any disjoint union of diabolos connecting S_0 and S_m satisfies $|S_i| \leq |S_0| + |S_m|$ for each i.

3. If the multi-stage graph corresponding to a sequence proving $A \equiv_d^\alpha B$ is a diabolo, then we can obtain another such sequence whose length will be at most $3(|A| + |B| - 2) + 1$.

Proof. (1) Obvious. (2) We could think that this is an immediate consequence of 1, but this is not always the case. Let us consider the disjoint union S of two three-stages (degenerated) diabolos S^1 and S^2, with $|S_0^1| = 1 = |S_2^2|$; $|S_0^2| = 8 = |S_2^1|$; $|S_1^1| = 5 = |S_1^2|$. Then we have $S_0 = S_0^1 \cup S_0^2$, $S_1 = S_1^1 \cup S_1^2$, $S_2 = S_2^1 \cup S_2^2$ and therefore $|S_1| = 10$, but $|S_0| = |S_2| = 9$. As a consequence, the result would be wrong if we put *max* instead of $+$. However, what we asserted is true in general: Let S the disjoint union of a family of diabolos S^j with $j \in \{1..k\}$, then we have $S_i = \bigcup_{j=1}^k S_i^j$, and as a consequence of 1, $|S_i^j| \leq |S_0^j| + |S_m^j|$, and therefore $|S_i| = \sum_{j=1}^k |S_i^j| \leq \sum_{j=1}^k |S_0^j| + \sum_{j=1}^k |S_m^j| = |S_0| + |S_m|$. (3) Whenever we have two relabeling steps at the same place, with no idempotency steps between them affecting that place, we can join them into a single relabeling step, without increasing the cost of the full transformation, because of triangular transitivity. As a consequence, we would have at most two relabeling steps for each idempotency step, and we have exactly $|A| + |B| - 2$ idempotency steps at each diabolo. So we have, at the moment, at most $3(|A| + |B| - 2)$ steps; possibly, we will need only one more relabeling step at the center of the diabolo. This gives us the bound $3(|A| + |B| - 2) + 1$. □

Corollary 3. *If we can prove $A \equiv_d^\alpha B$, then we can do it by means of a sequence whose intermediate sets satisfy $|S_i| \leq |A| + |B|$, and has at most $3(|A| + |B| - 2) + 1$ steps.*

Proof. We apply Th. 2 to reduce the multi-stage graph corresponding to the sequence proving $A \equiv_d^\alpha B$. Then apply Prop. 10 to get the bounds for the values $|S_i|$, and that for the number of steps. □

5.2 Now we go down into the second level

Once we have the base case of an inductive proof, we would like to proceed with the inductive case. When we thought that we had it, we decided to present first the particular case of the trees with only two levels, because its (bigger) simplicity would help the readers to understand the quite involved proof for the general case. Next we present the proof for this particular case. Starting from $p = \sum_{i\in I} a_i p_i$ and $q = \sum_{j\in J} b_j q_j$, once we have proved our Cor. 3, we can assume that the first level steps in the sequences proving $\pi_n(p) \equiv_d^\alpha \pi_n(q)$ are always the same and satisfy the bounds in the statement of Th. 2. In order to study in detail the second level steps in these sequences, we need to see how the summands p_i evolve along those sequences.

Once we had some bound for the width of any process p' obtained by the evolution of the summands p_i, and another one for the length of the subsequences producing their evolution, adding all these bounds and that corresponding to the first level, we would obtain the bound for the two first levels together. The following Prop. 11 proves a preliminary result. It says that whenever we have a sequence proving $p \equiv_d^\alpha q$ with a "limited number" of first level steps, but q contains summands aq_i where $||q_i||$ is "very large", then we can prune these summands, getting some q' for which $p \equiv_d^\alpha q'$ can be proved by means of a sequence that only contains processes p^i for which $||p^i||_2$ is "moderately large", and for any process q'' "between" q' and q (that means that q'' can be obtained by adding some branches to some subprocesses of q', and also q can be obtained from q'' in this way) we also have $p \equiv_d^\alpha q''$. Again, in order to make easier the comprehension of the proposition, we first present a lemma that covers the particular case corresponding to intermediate stages of a sequence connecting two processes p and q with $||p||$ and $||q||$ "small".

Lemma 3. *For any intermediate process p^i in the sequence $\mathscr{C}$ proving $p = \sum_{i=1}^n a_i p_i \equiv_d^\alpha q = \sum_{j=1}^m b_j q_j$, we can decompose p^i into $p^i = p^{i1} + p^{i2}$ in such a way that $||p^{i1}|| \leq ||p|| + ||q||$, and we have a sequence $\mathscr{C}' := p \equiv_{d_1}^\alpha p^{i1} \equiv_{d_2}^\alpha q$ with $d = d_1 + d_2$, which has at most $3(|I| + |J| - 2) + 1$ first level steps, and only uses intermediate processes $r = \sum_{k\in K} c_k r_k$ with $||r|| \leq ||p|| + ||q||$.*
Moreover, for any decomposition $p^{i2} = p^{i3} + p^{i4}$, we can obtain a sequence $\mathscr{C}'' := p \equiv_{d_1}^\alpha p^{i1} + p^{i3} \equiv_{d_2}^\alpha q$ which has at most $(3(|I| + |J| - 2) + 1) * (||p^{i3}|| + 1)$ first level steps, and only uses intermediate processes $r = \sum_{k\in K} c_k r_k$, with $||r|| \leq ||p|| + ||q|| + ||p^{i3}||$.

Proof. The first part is in fact a new formulation of Th. 2, observing that each node at the i-th stage of the multi-stage graph induced by $\mathscr{C}$ corresponds to a summand of the corresponding process p^i. When reducing the multi-stage graph, we are pruning some of these summands. Therefore, the obtained intermediate processes at the sequence $\mathscr{C}'$ are the processes p^{i1} of the searched decomposition of p^i. Then, the remaining summands in p^{i2} correspond to the nodes from the i-th stage of the multi-stage graph that were removed when reducing it. It is easy to see that any of these summands can be "reset" into the multi-stage graph by means of a path that will connect the corresponding node with the two sides of the original multi-stage graph. This requires $(3(|I| + |J| - 2) + 1)$ additional arcs, and therefore $(3(|I| + |J| - 2) + 1)$ more steps in the sequence $\mathscr{C}''$, while the width of the intermediate processes in it is increased at most by 1, when adding each of those paths. □

Proposition 11. *For each $f \in \mathbb{N}$, there is a constant $C_{2,f}$ such that if we have a sequence $\mathscr{C}$ proving $p \equiv_d^\alpha r$ with f first level steps and $r = \sum_{k\in K} c_k r_k$, with $r_k = \sum_{l\in K_k} c_{k,l} r_{k,l}$, then we can obtain some $r' = \sum_{k\in K} c_k r'_k$ with $r'_k = \sum_{l\in L_k} c_{k,l} r'_{k,l}$, where $L_k \subseteq K_k$, with $|L_k| \leq 2^f ||p||$, whose intermediate processes*

p^i are of the form $p^i = \sum a^i_j p^i_j$, with $p^i_j = \sum_{k\in K^i_j} b_k p^i_{j,k}$ and $|K^i_j| \leq 2^f||p||$, and $length(S') \leq C_{2,f}$. Moreover, taking $m \in \mathbb{N}$, we also have a family of constants $C_{2,f,m}$ such that for any $r'' = \sum c_k r''_k$ with $r''_k = \sum_{l\in L'_k} c_{k,l} r_{k,l}$, where $L_k \subseteq L'_k \subseteq K_k$, we can also prove $p \equiv^\alpha_d r''$, by means of a sequence $\mathscr{C}''$ whose intermediate processes p^i satisfy $||p^i||_2 \leq max\{2^f||p||, |L'_k|_{k\in K}\}$ and $length(S'') \leq C_{2,f,max\{|L'_k|_{k\in K}\}}$.

Theorem 3. *For all $k \in \{1,2\}$ and $w \in \mathbb{N}$ there exists a bound $lb(k,w) \in \mathbb{N}$ such that for all p,q with $||p||_k$, $||q||_k \leq w$ and $p \equiv^\alpha_d q$ we can prove the latter by means of a sequence $\mathscr{C}'$ that has no more than $lb(k,w)$ steps in each of its two first levels.*

Proof. At the same time that the existence of the bound $lb(k,w)$ we will prove a bound $wb(k,w)$ for the width $||p^i||_2$ of any process p^i along the 2-unfolding of the sequence $\mathscr{C}'$.
k=1 Th. 2 just states the result for this case, taking $wb(1,w) = 2w$ and $lb(1,w) \leq 6w$.
k=2 Let $\mathscr{C}$ be a sequence proving $p \equiv^\alpha_d q$ which has less than $lb(1,w)$ steps at the first level. If $\mathscr{C}$ contains no first level step, then we have $p = \sum_{i=1}^n a_i p_i$, $q = \sum_{i=1}^n a_i q_i$ and $\mathscr{C}$ can be factorized into a collection of sequences $(\mathscr{C}'_i)_{i=1}^n$ proving $p_i \equiv_{d_i} q_i$. Then, we only need to apply Th. 2 to each of these sequences, so that we could take $wb(2,w) = 2w$ and $lb(2,w) = 2w * 6w = 12w^2$.
If $\mathscr{C}$ contains some first level step, we select any of them and divide the 2-unfolding of $\mathscr{C}$ into $\mathscr{C}^1 \circ \langle s \rangle \circ \mathscr{C}^2$, where by abuse of notation we identify the sequences and their 1-unfolding. Let us consider the case in which the central step corresponds to a relabeling $ap'_i \to bp'_i$ (the other cases are analogous). Assume that $\mathscr{C}^1$ proves $p \equiv^\alpha_{d_1} p'$ and $\mathscr{C}^2$ proves $q' \equiv^\alpha_{d_3} q$, with $d = d_1 + \mathbf{d}(a,b) + d_3$. We can apply Prop. 11 to both sequences, taking $\mathscr{C}^2$ in the opposite direction, getting r' and r'', with $\mathscr{C}'^1$ proving $p \equiv^\alpha_{d_1} r'$ and $\mathscr{C}'^2$ proving $r'' \equiv^\alpha_{d_3} q$, that have the same steps at the first level than $\mathscr{C}^1$ and $\mathscr{C}^2$, respectively, and only use intermediate processes p^i with $||p^i||_2 \leq 2^{lb(1,w)-1} * w$. The result is also valid for any intermediate process between r_1 and p', and anyone between r_2 and q', but increasing the bounds in the adequate way. In particular, for the process $r_1 + r_2$, we have $p \equiv^\alpha_{d_1} r_1 + r_2$ and $r_1 + r_2 \equiv^\alpha_{d_3} q$, by means of two sequences that only include intermediate processes p^i with $||p^i||_2 \leq 2^{lb(1,w)} * w$. Joining these two sequences with the step s we obtain the sequence $\mathscr{C}'$. Adding the bounds for the corresponding lengths of the 2-unfolding of the two sequences obtained by application of Prop. 11, we obtain the bound $lb(2,w)$. To be more precise, the definitive value of $lb(2,w)$ will be the maximum of the bounds for the different cases (there are finitely many) considered above. □

Certainly, the proof is quite involved and difficult to follow, but it works in this case. Unfortunately, when we tried to develop it for the general case we discovered that some details failed, or at least cannot be justified in the same way. Let us roughly explain why: the argument used in Lemma 3 looks for an small "kernel" p^{i1} of any intermediate process along a sequence. Then Prop. 11 joins the kernels obtained when considering the two subsequences starting at any side of the original sequence and ending at any intermediate process. The second part of Lemma 3 says us that we can get in this way a sequence satisfying the desired bounds. But when we are in a deeper level, a duplication (by using idempotency) at a lower level could cause an undesired increasing of the "closure" of the union of the two (one from each side) kernels. And this problem could appear over and over (at least, we are not able to prove that this is not the case). Therefore, we have to leave this problem open, at the moment, any help?

6 Conclusions and future work

The first two authors of this paper are involved in a detailed study of bisimulation distances, that will constitute in fact the forthcoming Ph.D. Thesis of the first one, supervised by the second. As we have

already pointed out, while the global bisimulation distance we propose is more difficult to manage than the rest of bisimulation distances, we claim that it provides a much more accurate measure of the differences between two processes. Moreover, our approach has many nice properties, and we still think that continuity with respect to projections is one of them. Even if we have not been able yet to present a full continuity proof, we think that, after having invested a great number of hours in this quest, it is time to present here all our (partial) results until the moment.

We are now working on several extensions of our approach. In particular the modal interface framework [10, 11, 14] is a quite suggestive one, where we are obtaining quite promising results. We are also interested in stochastic distances [9] and those based on logics [6]. When considering probabilistic choices between branches associated with the same action, these two notions of distance *do* capture global differences. However, this is not the case when choices between different actions are considered. It is our intention to provide a "fully global" probabilistic distance (covering also the latter case) that will extend our global distance to these frameworks. Finally, we recently started to consider interval temporal logics [5], aiming at devising a suitable notion of global distance between interval models.

References

[1] P. Cerný, T. A. Henzinger, and A. Radhakrishna. Quantitative simulation games. In *EMAP 2010*, volume 6200 of *LNCS*, pages 42–60. Springer, 2010.

[2] P. Cerný, T. A. Henzinger, and A. Radhakrishna. Simulation distances. In *CONCUR 2010*, volume 6269 of *LNCS*, pages 253–268. Springer, 2010.

[3] L. de Alfaro, T. A. Henzinger, and R. Majumdar. Discounting the future in systems theory. In *ICALP 2003*, volume 2719 of *LNCS*, pages 1022–1037. Springer, 2003.

[4] L. de Alfaro, R. Majumdar, V. Raman, and M. Stoelinga. Game relations and metrics. In *LICS 2007*, pages 99–108. IEEE Computer Society, 2007.

[5] D. Della Monica, V. Goranko, A. Montanari, and G. Sciavicco. Interval Temporal Logics: a Journey. *Bulletin of the EATCS*, 105, October 2011.

[6] J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden. Metrics for labelled markov processes. *Theoretical Computer Science*, 318(3):323–354, 2004.

[7] J. Desharnais, F. Laviolette, and M. Tracol. Approximate analysis of probabilistic processes: Logic, simulation and games. In *QEST 2008*, pages 264–273. IEEE Computer Society, 2008.

[8] U. Fahrenberg, C. R. Thrane, and K. G. Larsen. Distances for weighted transition systems: Games and properties. In *QAPL 2011*, pages 134–147, 2011.

[9] N. Ferns, P. Panangaden, and D. Precup. Bisimulation metrics for continuous markov decision processes. *SIAM J. Comput.*, 40(6):1662–1714, 2011.

[10] K. G. Larsen, U. Nyman, and A. Wasowski. Modal I/O automata for interface and product line theories. In *ESOP 2007*, volume 4421 of *LNCS*, pages 64–79. Springer, 2007.

[11] G. Lüttgen and W. Vogler. Modal interface automata. *Logical Methods in Computer Science*, 9(3), 2013.

[12] R. Milner. *Communication and concurrency*. Prentice Hall, 1989.

[13] D.M.R. Park. Concurrency and automata on infinite sequences. In *Theoretical Computer Science, 5th Gl-Conference*, volume 104 of *LNCS*, pages 167–183. Springer, 1981.

[14] J.-B. Raclet, E. Badouel, A. Benveniste, B.ît Caillaud, A. Legay, and R. Passerone. A modal interface theory for component-based design. *Fundam. Inform.*, 108(1-2):119–149, 2011.

[15] D. Romero-Hernández and D. de Frutos-Escrig. Defining distances for all process semantics. In *FMOODS/FORTE 2012*, volume 7273 of *LNCS*, pages 169–185. Springer, 2012.

[16] D. Romero-Hernández and D. de Frutos-Escrig. Distances between processes: A pure algebraic approach. In *WADT 2012*, volume 7841 of *LNCS*, pages 265–282. Springer, 2012.

[17] D. Romero-Hernández and D. de Frutos-Escrig. Coinductive definition of distances between processes: Beyond bisimulation distances. In *FORTE 2014*, volume 8461 of *LNCS*, pages 249–265. Springer, 2014.

[18] Davide Sangiorgi. On the origins of bisimulation and coinduction. *ACM Trans. Program. Lang. Syst.*, 31(4), 2009.

[19] C. Stirling. The joys of bisimulation. In *MFCS 1998*, volume 1450 of *LNCS*, pages 142–151. Springer, 1998.

[20] F. van Breugel. A behavioural pseudometric for metric labelled transition systems. In *CONCUR 2005*, volume 3653 of *LNCS*, pages 141–155. Springer, 2005.

[21] F. van Breugel and J. Worrell. A behavioural pseudometric for probabilistic transition systems. *Theoretical Computer Science*, 331(1):115–142, 2005.

Part III

Appendix

More about our publications

"... we shall see if there is something written in this diary that will allow us to investigate and learn what we wish to know."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter 23 Part-1

"... veremos si en este librillo de memoria hay alguna cosa escrita por donde podamos rastrear y venir en conocimiento de lo que deseamos."

This chapter is devoted to present some missed proofs and other additional material that, due to space reasons, we had to omit when preparing the final versions sent to the forums where the papers included in this thesis were published. We will review each of our publications in a different section. Furthermore, we include at the beginning of each section, some small but important bugs that unfortunately we did not noticed until the papers were published and could obscure your initial comprehension when reading the papers. It is possible that we have not detected here all the bugs, but despite this we think that, after our multiple readings, the papers have improved their quality.

8.1 On the unification of process semantics: logical semantics

Some bugs found

pg. 54 line 6: ... consider a logic $\mathcal{L}'_Z$ and the index ...
pg. 58 line 13: ... simply as: $\mathcal{N}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}''_\mathcal{N}) = \mathcal{N}(\mathcal{L}''_\mathcal{N}) \bigcap \mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$ where $\mathcal{L}''_N$ is the set ...

Additional proofs

Theorem 1 For all $N \in \{U, C, I, T, S\}$ and any two processes p and q, $p \sqsubseteq_{NS} q$ iff $p \leq^b_N q$.
Proof. See the proof of Theorem 4.9 page 26 in [dFGPR13]. □

Theorem 2 (1) $p \sqsubseteq_{RT} q$ iff $p \leq^l_I q$; (2) $p \sqsubseteq_{FT} q$ iff $p \leq^{l\supseteq}_I q$; (3) $p \sqsubseteq_R q$ iff $p \leq^{lf}_I q$; (4) $p \sqsubseteq_F q$ iff $p \leq^{lf\supseteq}_I q$.
Proof. (1) See the proof of Proposition 4.15 (2) page 27 in [dFGPR13].
(2) See the proof of Proposition 4.18 page 28 in [dFGPR13].
(3) Since the definition of $\leq^{lf}_I$ ignores all the intermediate ready sets X_i with $i < n$ and requires the final ready sets to coincide, it is obvious that it defines the readiness preorder.
(4) See the proof of Proposition 4.18 page 28 in [dFGPR13]. □

Proposition 2 1. $\mathcal{L}'_{RS} \supseteq \mathcal{L}_{RS}$. We also have $\mathcal{L}_{RS} \subsetneq \mathcal{L}'_{RS}$.
2. $\mathcal{L}'_{RT} \supseteq \mathcal{L}_{RT}$. We also have $\mathcal{L}_{RT} \subsetneq \mathcal{L}'_{RT}$.
3. $\mathcal{L}'_{FT} \supseteq$ desugared($\mathcal{L}_{FT}$), where the desugaring function removes the syntactic sugar used in $\mathcal{L}_{FT}$.
4. $\mathcal{L}'_R \supseteq \mathcal{L}_R$. We also have $\mathcal{L}_R \subsetneq \mathcal{L}'_R$.
5. $\mathcal{L}'_F \supseteq$ desugared($\mathcal{L}_F$), where the desugared function removes the syntactic

sugar used in $\mathcal{L}_F$.
Proof. See the proof of Proposition 6.5 page 45 in [dFGPR13]. □

Proposition 3 We have (1) $\mathcal{L}_{RS} \sim \mathcal{L}'_{RS}$; (2) $\mathcal{L}_{RT} \sim \mathcal{L}'_{RT}$; (3) $\mathcal{L}_{FT} \sim \mathcal{L}'_{FT}$; (4) $\mathcal{L}_R \sim \mathcal{L}'_R$ and (5) $\mathcal{L}_F \sim \mathcal{L}'_F$.
Proof. See the proof of Proposition 6.6 page 46 in [dFGPR13]. □

Theorem 3 1. The logical semantics $\sqsubseteq'_{RS}$ induced by the logic $\mathcal{L}'_{RS}$ is equivalent to the observational branching semantics defined by $\leq^b_I$, generated by the set of branching general observations BGO_I.
2. The logical semantics $\sqsubseteq'_{RT}$ (resp. $\sqsubseteq'_{FT}$, $\sqsubseteq'_R$, $\sqsubseteq'_F$) induced by the logic $\mathcal{L}'_{RT}$ (resp. $\mathcal{L}'_{FT}$, $\mathcal{L}'_R$, $\mathcal{L}'_F$) is equivalent to the observational linear semantics defined by the domain of linear general observations LGO_I, ordered by $\leq^l_I$ (resp. $\leq^{l\supseteq}_I$, $\leq^{lf}_I$, $\leq^{lf\supseteq}_I$,) defined at Def. 8.
Proof. See the proof of Theorem 6.7 page 46 in [dFGPR13]. □

Proposition 4 We have (1) $\mathcal{L}'_S \sim \mathcal{L}_S$, (2) $\mathcal{L}'_{CS} \sim \mathcal{L}_{CS}$ and (3) $\mathcal{L}'_{2S} \sim \mathcal{L}_{2S}$.
Proof. See the proof of Proposition 6.11 page 50 in [dFGPR13]. □

Proposition 6 We have (1) $\mathcal{L}'_{\leq^{lf}_U} = \mathcal{L}'_{\leq^{l}_U} = \mathcal{L}'_{\leq^{l\supseteq}_U} = \mathcal{L}'_{\leq^{l\subseteq}_U} = \mathcal{L}'_{\leq^{lf\supseteq}_U} = \mathcal{L}'_{\leq^{lf\subseteq}_U} = \mathcal{L}_T$ and (2) $\mathcal{L}'_{\leq^{lf\supseteq}_C} = \mathcal{L}'_{\leq^{lf\subseteq}_C} = \mathcal{L}'_{\leq^{l\supseteq}_C} = \mathcal{L}'_{\leq^{l\subseteq}_C} = \mathcal{L}'_{\leq^{lf}_C} = \mathcal{L}'_{\leq^{l}_C} = \mathcal{L}_{CT}$.
Proof. See the proof of Proposition 6.18 page 53 in [dFGPR13]. □

Proposition 7 We have $\mathcal{L}'_{\leq^{lf}_T} = \mathcal{L}_{PF}$.
Proof. See the proof of Proposition 6.20 page 53 in [dFGPR13]. □

Proposition 8 We have $\mathcal{L}'_{D_I} \supseteq \mathcal{L}_{PW}$.
Proof. See the proof of Proposition 6.23 page 54 in [dFGPR13]. □

Theorem 4 Each set of normal formulas $\mathcal{NY}_{\mathcal{N}}(\mathcal{L}''_{\mathcal{N}})$ associated to any of the semantics in the spectrum is equivalent to the corresponding full set of formulas $\mathcal{L}'_{\mathcal{Y}_{\mathcal{N}}}$.
Proof. See the proof of Theorem 7.3 page 56 in [dFGPR13]. □

Theorem 5 If we restrict ourselves to finite image processes, any complete normal formula $\varphi \in \mathcal{CN}(\mathcal{L})$ can be approximated by a set of finite normal formulas

$\{\varphi^k \mid k \in \mathbb{N}\}$ that only use finite conjunction, that is, we have $p \models \varphi \Leftrightarrow p \models \varphi^k$ $\forall k \in \mathbb{N}$.

Proof. See the proof of Theorem 7.5 page 56 in [dFGPR13]. □

Theorem 6 We can define a natural correspondence between the set of complete normal formulas associated to a semantics $\mathcal{CN}_{\mathcal{Y}_\mathcal{N}}(\mathcal{L}''_\mathcal{N})$ and the corresponding domain of observations BGO_N or LGO_N. That correspondence $\leftrightarrow$ satisfies that $\varphi \leftrightarrow \theta \Rightarrow (p \models \varphi \Leftrightarrow \theta \in XGO_N(p))$ with $X = B$ or $X = L$. Moreover, this correspondence produces the following results for each of the semantics in the spectrum:

1. The set of complete normal formulas $\mathcal{CN}_{\mathcal{NS}}(\mathcal{L}''_\mathcal{N})$ (resp. $\mathcal{CN}_{\mathcal{D}_\mathcal{N}}(\mathcal{L}''_\mathcal{N})$) and the domain of branching general observations GBO_N (resp. $dBGO_N$) are isomorphic, that is, $\leftrightarrow$ is one to one.

2. The set of complete normal formulas $\mathcal{CN}_{\leq^l_N}(\mathcal{L}''_\mathcal{N})$, $\mathcal{CN}_{\leq^{l\supseteq}_N}(\mathcal{L}''_\mathcal{N})$ and the domain of linear general observations LGO_N are isomorphic, that is, $\leftrightarrow$ is one to one.

3. The set of complete normal formulas $\mathcal{CN}_{\leq^{lf}_N}(\mathcal{L}''_\mathcal{N})$ (resp. $\mathcal{CN}_{\leq^{lf\supseteq}_N}(\mathcal{L}''_\mathcal{N})$) and the quotient domain $LGO_N/_{\simeq^{lf}_N}$ (resp. $LGO_N/_{\simeq^{lf\supseteq}_N}$) are isomorphic, that is, $\leftrightarrow^{-1}$ is injective and $\varphi \leftrightarrow \theta$ iff $\theta \simeq^{lf\supseteq}_N \theta_\varphi$, for some adequate θ_φ.

Proof. See the proof of Theorem 7.8 page 57 in [dFGPR13]. □

Theorem 7 The logical semantics $\sqsubseteq'_{\mathcal{Y}_\mathcal{N}}$ induced by the logic $\mathcal{L}'_{\mathcal{Y}_\mathcal{N}}$, where $\mathcal{Y}_\mathcal{N} \in \{NS, \leq^l_N, \leq^{l\supseteq}_N, \leq^{lf}_N, \leq^{lf\supseteq}_N, D_N\}$, is equivalent to the corresponding observational semantics, defined at Def. 6 and Def. 7.

Proof. See the proof of Theorem 7.10 page 58 in [dFGPR13]. □

8.2 Unifying the Linear time-Branching time spectrum of strong process semantics

Some bugs found

pg. 44 in Def. 6.3: All the $\mathcal{L}'_I$ should be read $\mathcal{L}_I$. All the $\bigwedge_{a\in X_1} a\top$ should be read $\bigwedge_{a\top\in X_1} a\top$. All the $\bigwedge_{b\in X_2} b\top$ should be read $\bigwedge_{b\top\in X_2} b\top$.

pg. 45 line -10: ... To prove that $\mathcal{L}_{RT} \subseteq \mathcal{L}'_{RT}$, it is sufficient to ...

pg. 46 line 9: Any conjunction of formulas in $\mathcal{L}_I$ and negation of formulas in it, can be obtained as the disjunction of the formulas X in $\mathcal{L}_{RS}$ describing ...

pg. 46 line -9: ... when we consider a logic $\mathcal{L}'_Z$ and the index ...

pg. 47 ln -3: ... which is analogous to the case for $\mathcal{N}(\mathcal{L}_{\mathcal{RS}})$ and $\mathcal{CN}(\mathcal{L}_{\mathcal{RS}})$ above.

pg. 50 ln -5: ... two trivial formulas $\top$ and $\neg\top$ because $\mathcal{L}'_U = \{\top\}$.

8.3 Defining distances for all process semantics

Some bugs found

Along the paper sons should be read children.

pg. 172 line -11: ... $\overline{d}(a,b) = 0 \Leftrightarrow a = b$, $\overline{d}(a,c) + \overline{d}(c,b) \geq \overline{d}(a,b)$ $\forall a, b, c \in Act$. Later, ...

pg. 176 line 7: Therefore, we have $d^S_{\overline{d}}(q,p) \leq \overline{d}(c,d) = 1$. Next we see ...

pg. 176 line 12: ... which produces $d_{\overline{d}}(p,q) \leq \overline{d}(b,c) = 1$.

Additional proofs

Theorem 1 $p \sqsubseteq_S q$ (resp. $p \not\sqsubseteq_S q$) if and only if $\mathbb{D}$ (resp. $\mathbb{A}$) has a winning strategy for the simulation game starting at (p,q).
Proof. $\underline{\Rightarrow |}$ The simulation game proceeds from the configuration (p,q) as follows:

- $\mathbb{A}$ chooses a transition in $L_1 : p \xrightarrow{a} p'$.
- $\mathbb{D}$ must execute the same action in the other side: $q \xrightarrow{a} q'$. This would be possible since by hypothesis $p \sqsubseteq_S q$.
- The game proceeds in the same way from (p', q').

As far as by hypothesis $p \sqsubseteq_S q$ the defender can choose q' in such a way that $p' \sqsubseteq_S q'$. And repeating the reasoning starting from the configuration (p', q') we will continue defining the strategy that makes $\mathbb{D}$ to win the game.

$\underline{\Leftarrow |}$ We prove that the relation $\mathcal{R} = \{(p,q) \mid \mathbb{D}$ has a winning strategy when playing the simulation game starting from $(p,q)\}$ is indeed a simulation. By definition of winning strategy, whenever $\mathbb{A}$ plays $p \xrightarrow{a} p'$, $\mathbb{D}$ must be able to reply with some $q \xrightarrow{a} q'$, so that he will continue winning the game. Therefore,

$(p', q') \in \mathcal{R}$ as required.

Since any play has a winner, the statements in brackets immediately follows from those proved above. □

Theorem 3 $p \sim q$ (resp. $p \not\sim q$) if and only if $\mathbb{D}$ (resp. $\mathbb{A}$) has a winning strategy for the bisimulation game starting at (p, q).
Proof. Analogous to the proof of Theorem 1. □

Theorem 4 1. $(H^B_n)_{n\in\mathbb{N}}$ is a cbdf.
2. If $(R_n)_{n\in\mathbb{N}}$ is a cbdf then $R_n \subseteq H^B_n$.
Proof.

– 1| We prove that $(H^B_n)_{n\in\mathbb{N}}$ satisfies the definition of *cbdf*, by rule induction on the definition of H^B_n. In the diagrammatic proofs below we only present for each of the cases one of the several symmetric instances that we have to check.

$$
\begin{array}{lcccl}
(1): & p & H^B_n & q & \\
 & \forall a \downarrow & & \downarrow \exists\ b{=}a & (\overset{df}{\Leftarrow} p \sim q) \\
 & p' & H^B_n & q' & (\overset{(1)}{\Leftarrow} p' \sim q')
\end{array}
$$

$$
\begin{array}{lcc}
(2): & ap \quad H^B_{n+\overline{d}(a,b)} \quad bq & \\
 & a\downarrow \qquad\qquad \downarrow b & \\
 & pH^B_{n'+\overline{d}(a,b)-\overline{d}(a,b)}q & \\
 & \Downarrow & \\
 & p\ H^B_n\ q &
\end{array}
\qquad
\begin{array}{lcc}
(3): & p+q \quad H^B_n \quad p'+q' & \\
 & a\downarrow \qquad\qquad \downarrow b & \\
 & p'' \quad H^B_{n'-\overline{d}(a,b)} \quad p''' & \\
 & \Uparrow pH^B_n p' \wedge qH^B_n q' \ with\ n{\geq}n' & \\
 & p \qquad H^B_n \qquad p' & \\
 & a\downarrow \qquad\qquad \downarrow b \quad (by \quad i.h.) & \\
 & p'' \quad H^B_{n-\overline{d}(a,b)} \quad p''' &
\end{array}
$$

$$
\begin{array}{lcccl}
(4): & p & H^B_{n+m} & r & \\
 & a\downarrow & & \downarrow b & (\Leftarrow_{i.h.(4)}) \\
 & p'H^B_{n+m-(\overline{d}(a,c)+\overline{d}(c,b))} & & r' & \\
 & \Downarrow \overline{d}(b,a)\leq\overline{d}(c,b)+\overline{d}(a,c) & & & \\
 & p' \quad H^B_{n+m-\overline{d}(b,a)} & & r' &
\end{array}
\qquad
\begin{array}{lcccccl}
 & p & H^B_n & q & q & H^B_m & r \\
 & a\downarrow & & \downarrow c & c\downarrow & & \downarrow b \\
 & p'H^B_{n-\overline{d}(a,c)} & & q' & q'H^B_{m-\overline{d}(c,b)} & & r'
\end{array}
$$

– 2| We use complete induction on the depth of p. Let $p = \sum ap_a$ and $q = \sum bq_b$ such that

$$\begin{array}{ccccc} p & R_n & q & & p & R_n & q \\ \forall\ a\downarrow & \Rightarrow & \downarrow\exists\ b & \wedge & \exists\ a'\downarrow & \Leftarrow & \downarrow\forall\ b \\ p_a & R_{n-\overline{d}(b,a)} & q_b & & p_{a'} & R_{n-\overline{d}(a',b)} & q_b \end{array}$$

Then

$$\left.\begin{array}{c} p_a\ R_{n-\overline{d}(b,a)}\ q_b \\ \wedge \\ p_{a'}\ R_{n-\overline{d}(a',b)}\ q_b \end{array}\right\} \overset{i.h.}{\Rightarrow} \left.\begin{array}{c} p_a\ H^B_{n-\overline{d}(b,a)}\ q_b \\ \wedge \\ p_{a'}\ H^B_{n-\overline{d}(a',b)}\ q_b \end{array}\right\} \overset{(2)}{\Rightarrow} \left.\begin{array}{c} ap_a\ H^B_n\ bq_b \\ \vee \\ a'p_{a'}\ H^B_n\ bq_b \end{array}\right\}$$

On the first line we have all the summands of p, while on the second we have all the summands of q. It is then immediate that if we add all of them we can apply (3′) as many times as needed to derive $p\,H^B_n\,q$ as required. □

Proposition 13 Whenever we have two semantics $\mathcal{L}_1$ and $\mathcal{L}_2$ and the first is finer than the latter ($\sqsubseteq_{\mathcal{L}_1} \subseteq \sqsubseteq_{\mathcal{L}_2}$), we also have $gd^{\mathcal{L}_1}_{\overline{d}}(p,q) \leq n \Rightarrow gd^{\mathcal{L}_2}_{\overline{d}}(p,q) \leq n$, for all processes p, q and any value $n \in \mathbb{N}$.

Proof. From $gd^{\mathcal{L}_1}_{\overline{d}}(p,q) \leq n$ we can infer $pG^{\mathcal{L}_2}_n q$ by applying our distance rules in Def. 10. We will proceed by structural induction on the depth of processes when necessary.

1. By hypothesis, we have $p \sqsubseteq_{\mathcal{L}_1} q$. Then since $\mathcal{L}_1$ if finer than $\mathcal{L}_2$ we have by definition that $p \sqsubseteq_{\mathcal{L}_2} q$, thus proving $pG^{\mathcal{L}_2}_n q$.
2. By hypothesis, we can derive $apG^{\mathcal{L}_1}_{n+\overline{d}(b,a)} bq$ from $pG^{\mathcal{L}_1}_n q$. Now by applying i.h. we have $pG^{\mathcal{L}_2}_n q$, so by applying rule 2 we have $apG^{\mathcal{L}_2}_{n+\overline{d}(b,a)} bq$.
3. By hypothesis, we can derive $p + qG^{\mathcal{L}_1}_n p' + q$ from $pG^{\mathcal{L}_1}_n p'$. Now by applying our i.h. we have $pG^{\mathcal{L}_2}_n p'$, so by applying rule 3 we have $p + qG^{\mathcal{L}_2}_n p' + q$.
4. By hypothesis, we can derive $pG^{\mathcal{L}_1}_{n+n'} r$ from $pG^{\mathcal{L}_1}_n q$ and $qG^{\mathcal{L}_1}_{n'} r$. Now by applying i.h. we have $pG^{\mathcal{L}_2}_n q$ and $qG^{\mathcal{L}_2}_{n'} r$, so by applying rule 4 we have $pG^{\mathcal{L}_2}_{n+n'} r$.

We will have $gd^{\mathcal{L}_2}_{\overline{d}}(p,q) \leq n$ □

8.4 Distances between processes: a pure algebraic approach

Additional proofs

Proposition 1 If $E_{\mathcal{D}}$ is a system that only contains equations on $\equiv_0$, then the system $\mathbf{dDED}(E_{\mathcal{D}})$ is "essentially" equivalent to $\mathbf{DED}(E)$, where $E = \{t \equiv t' \mid t \equiv_0 t' \in E_{\mathcal{D}}\}$. This means that $\vdash_{E_{\mathcal{D}}} t \equiv_d t' \Leftrightarrow \vdash_E t \equiv t', \ \forall d \in \mathcal{D}$.
Proof. $\Leftarrow|$ This is an immediate consequence of the fact that $\equiv_0$ just reflects the quotient algebra on top of which we will define our distance relations.
$\Rightarrow|$ By a simple induction on the derivation of $t \equiv_d t'$: any derivation of such a pair can be transformed into a derivation of $t \equiv_0 t'$ which corresponds to a derivation of $t \equiv t'$ under $\vdash_E$, since $E = \{t \equiv t' \mid t \equiv_0 t' \in E_{\mathcal{D}}\}$. □

Proposition 2 Given a system of distance equations $E_{\mathcal{D}}$, if we define $E^0_{\mathcal{D}} = \{t \equiv_d t' \mid d = 0\}$ and we consider the set of ordinary equations $E = \{t \equiv t' \mid t \equiv_0 t' \in E^0_{\mathcal{D}}\}$, then we can see the family of distance relations induced by $\vdash_{E_{\mathcal{D}}}$ as a family of distance relations between the equivalence classes induced by $\vdash_E$,

$$[t] \equiv_d [t'] ::= \vdash_{E_{\mathcal{D}}} t \equiv_d t'$$

Proof. By applying the triangular transitivity rule we have that our statement is correct.

$$\left.\begin{array}{c} [t_0] = [t_1] \ \Leftrightarrow t_0 \equiv_0 t_1 \\ \wedge \\ [t'_0] = [t'_1] \ \Leftrightarrow t'_0 \equiv_0 t'_1 \end{array}\right\} \ [t_0] \equiv_d [t_1] \ \Leftrightarrow \vdash_{E_{\mathcal{D}}} t_0 \equiv_d t'_1 \ \Leftrightarrow \vdash_E t_0 \equiv_d t'_1.$$

□

Proposition 3, from the appendix Defs. 14, 15, 16 are obviously equivalent.
Proof. Def. 14 $\Rightarrow$ Def. 15| By induction on the length n of the sequence that produces $p \rightsquigarrow_d p'$.

- n=1 Trivial.
- If we have $p \rightsquigarrow^1_{d_1} p_1 \rightsquigarrow^1_{d_2} p_2 \cdots \rightsquigarrow^1_{d_n} p_n = p'$ where $\sum_{i=1}^n d_i = d$. We can see this sequence as $p \rightsquigarrow^1_{d_1} p_1 \rightsquigarrow_{d^2} p'$ where $d^2 = \sum_{i=2}^n d_i$ and $p \rightsquigarrow_{d^2} p'$ is generated by a sequence of length n. Thus, by applying the i.h we have $p_1 \rightsquigarrow^*_{d_2} p'$, and then $p \rightsquigarrow^*_{d_1+d^2} p'$, that is $p \rightsquigarrow^*_d p'$.

Def. 15 ⇒ Def. 16| Since $p \rightsquigarrow^1_{d_1} p''$ is a particular case of $p \rightsquigarrow^*_{d_1} p''$, we immediately obtain the desired result.
Def. 16 ⇒ Def. 14| By rule induction on the application of Def. 16.

- If $p \mid\!\rightsquigarrow_d q$ is obtained by applying rule 1, we immediately have $p \rightsquigarrow_d q$ simply considering the sequence $p \rightsquigarrow^1_d p_1 = q$.
- If $p \mid\!\rightsquigarrow_{d_1} q$ is obtained by applying rule 2, by using two times the i.h. we get $p \rightsquigarrow^1_{d_1} p_1 \rightsquigarrow^1_{d_2} p_2 \rightsquigarrow^1_{d_3} \cdots \rightsquigarrow^1_{d_n} p_{n_1} = q$ and $q \rightsquigarrow^1_{d'_1} q_1 \rightsquigarrow^1_{d'_2} q_2 \rightsquigarrow^1_{d'_3} \cdots \rightsquigarrow^1_{d'_{n_2}} q_{n_2} = p\prime$ with $\sum_{i=1}^{n_1} d_{n_i} + \sum_{i=1}^{n_2} d'_{n_i} = d$. Putting both sequences one after the other, we obtain the sequence that proves $p \rightsquigarrow^1_d p'$. □

8.5 Coinductive definition of distances between processes: beyond bisimulation distances

Some bugs found

pg. 254 line 5: ... introduce a "discount factor" $\alpha \in (0, 1]$. Then, the differences ...

Additional proofs

Proposition 1 For any *lts* with initial state $(N, succ, n_0)$, and its unfolding $(\overline{N}, \overline{succ}, \overline{n_0})$, we have $n_0 \sim \overline{n_0}$.
Proof. We define $fold : \overline{N} \to N$ taking $fold(n_0 a_1 \dots n_i) = n_i$, and by definition of unfolding we have that $R = \{(\overline{n}, fold(\overline{n})\}$ is a bisimulation which contains the pair $(\overline{n_0}, n_0)$. □

Proposition 2 For any $t \in Tree(\mathbb{A})$ and $l, k \in \mathbb{N}$ with $l \leq k$, we have $\pi_l(\pi_k(t)) = \pi_l(t)$. Any finitary tree is defined by its sequence of projections: $\forall t, t' \in FyTrees(\mathbb{A})$ $(\forall k \in \mathbb{N}\ \pi_k(t) \sim \pi_k(t')) \Rightarrow t \sim t'$.
Proof. The first part asserting $\pi_l(\pi_k(t)) = \pi_l(t)$ is trivial by definition of the k-th cut or projection.
For the second part we prove that the relation $R = \{(t, t') \mid \pi_k(t) \sim \pi_k(t')\ \forall k \in \mathbb{N}\}$ is indeed a bisimulation. Let tRt', then we have $t = \sum_{i \in I} a_i t_i$, $t' = \sum_{j \in J} b_j t'_j$. In particular we have $t \xrightarrow{a_i} t'$, and then $\pi_k(t) \xrightarrow{a_i} \pi_{k-1}(t_i)\ \forall \geq 1$. So that from $\pi_k(t) \sim \pi_k(t')$ we conclude that $\forall k \geq 1\ \exists j_k \in J\ \pi_{k-1}(t_i) \sim \pi_{k-1}(t'_{j_k})$. But since

$|J| < \infty$ there must be infinitely many k's for which $j_k = j$ for some fixed value $j \in J$. Then, it is immediate to check that $\pi_{k-1}(t_i) \sim \pi_{k-1}(t_j)\ \forall k \geq 1$, hence $t_i R t_j$ as required in order to R will be a bisimulation relation. □

Proposition 3 The value of the quantitative game defining the "classical" bisimulation distance $dist^{\alpha}_{\mathbf{d}}(t, t')$ is $inf(\{d \in \mathbb{R}^+ \mid d^{\alpha}_{\mathbf{d}}(t, t') \leq d\})$.
Proof. First, note that the result is not immediate because the expression $d^{\alpha}_{\mathbf{d}}(t, t') \leq d$ on the rhs corresponds to the new notation introduced in Def. 6 and not to the comparison between the values of $dist^{\alpha}_{\mathbf{d}}(t, t')$, at the lhs, and d.

Now, by definition of the value of the game $dist^{\alpha}_{\mathbf{d}}(t, t')$, if we have $dist^{\alpha}_{\mathbf{d}}(t, t') \leq d$, for any strategy of the attacker we have another of the defender which allows him to lose no more than d at the game. This means that

$$\begin{array}{ccc} t & & t' \\ a\downarrow & \Longrightarrow & \downarrow b \\ t_1 & & t'_1 \end{array} \qquad with \quad dist^{\alpha}_{\mathbf{d}}(t_1, t'_1) < \tfrac{1}{\alpha}(d - \mathbf{d}(a, b))$$

from which we immediately obtain that $\{(t, t', d) \mid d > dist^{\alpha}_{\mathbf{d}}(t, t')\}$ is indeed a cbdf for d and α, from where the result follows. For the converse, it is clear that the fact $(t, t', d) \in R$ where R is a cbdf immediately produces a strategy for the defender which allows him to lose at most d at the game, from which the desired result follows. □

Theorem 2 For all t, $t' \in FyTrees(\mathbb{A})$ and any discount factor $\alpha \in (0, 1]$, we have $t \sim t'$ if and only if $d^{\alpha}_{\mathbf{d}}(t, t') \leq 0$, if and only if $dist^{\alpha}_{\mathbf{d}}(t, t') = 0$.
Proof. $\underline{\Rightarrow \mid}$ The implication $t \sim t' \Rightarrow d^{\alpha}_{\mathbf{d}}(t, t') \leq 0$, is immediate because any bisimulation $\mathcal{R}$ turns into the cbdf $\mathcal{R} = \{(t, t', 0) \mid t R t'\}$.
$\underline{\Leftarrow \mid}$ For the converse we use the same reasoning as in the proof of Prop. 2. Let us remark here that the classical bisimulation distance would produce "counterexamples" for this result when infinitary trees would be considered in two different scenarios. The first corresponds to the alphabets with a "continuous" distance, e.g. $\mathcal{A} = \{\frac{1}{n} \mid n \in \mathbb{N}\} \cup \{0\}$. Then, for the two (depth 1) trees $t = \sum_{n \in \mathbb{N}} \frac{1}{n}$ and $t' = \mathbf{0} + \sum_{n \in \mathbb{N}} \frac{1}{n}$, we clearly have $d^{\alpha}_{\mathbf{d}}(t, t') = 0$. It is interesting to note that instead we have not $d^{\alpha}_{\mathbf{d}}(t, t') \leq 0$, where we are using our notation in Def. 6. A similar result can be obtained for finite alphabets with a discrete distance, e.g. $\mathcal{A} = \{a, b\}$ with $\mathbf{d}(a, b) = 1$. Then, for the infinite depth trees $t = \sum_{n \geq 0} b^n a^{\infty}$ and $t' = \sum_{n \geq 0} b^n a^n$, we have again $d^{\alpha}_{\mathbf{d}}(t, t') = 0\ \forall \alpha < 1$, but we have not $d^{\alpha}_{\mathbf{d}}(t, t') \leq 0$

The second part of this theorem asserting $d^{\alpha}_{\mathbf{d}}(t,t') \leq 0$, if and only if $dist^{\alpha}_{\mathbf{d}}(t,t') = 0$ is trivial by using Prop. 3. □

Proposition 4 If $\mathcal{D}$ is an α-ccd, then $\mathcal{D}^*$ and $\mathcal{D}^+$ are too.
Proof.

- We prove that all triples $(t,t',d) \in \mathcal{D}^*$ satisfy the condition to be a ccd by induction on the number of times r that we need to apply ii) in Def. 10 to derive $(t,t',d) \in \mathcal{D}^*$.
  $r = 0|$ Then we have $(t,t',d) \in \mathcal{D}$, and since $\mathcal{D}$ is a ccd and $\mathcal{D} \subseteq \mathcal{D}^*$, the condition is satisfied.
  $r > 0|$ Let us consider $(t,t'',d+d') \in \mathcal{D}^*$ with $(t,t',d),(t',t'',d') \in \mathcal{D}^*$ derived by two shorter derivations. Then, by i.h. we have two sequences $t = t^{1,0} \equiv^{\mathcal{D},\alpha}_{d_1} \cdots \equiv^{\mathcal{D},\alpha}_{d_n} t^{1,n} = t'$ and $t' = t^{2,0} \equiv^{\mathcal{D},\alpha}_{d_1} \cdots \equiv^{\mathcal{D},\alpha}_{d_m} t^{2,m} = t''$ proving that those two triples satisfy the condition in order $\mathcal{D}^*$ to be a ccd. And their concatenation is clearly a proof of the fact that $(t,t'',d+d')$ satisfies the condition too.

- We prove that all triples $(t+t'',t'+t'',d) \in \mathcal{D}^+$ satisfy the condition to be a ccd. We only need to consider the sequence proving the condition for (t,t',d) in order to $\mathcal{D}$ was an α-ccd, then it is clear that adding t'' to all the trees along this sequence we obtain a sequence proving the condition for $(t+t'',t'+t'',d)$, in order $\mathcal{D}^+$ was a $+$-ccd.

□

Proposition 5 1.For $t,\ t' \in FyTrees(\mathbb{A}), \alpha \in (0,1]$, we have $t \sim t' \Leftrightarrow t \equiv^{\alpha}_0 t'$.
2.Our global bisimulation distance is greater or equal than the classical one.
Proof. $\Rightarrow|$ If $t \sim t'$ we can apply the two first rules in Def. 8, to obtain $t \equiv^{\alpha}_0 t + t' \equiv^{\alpha}_0 t'$, and applying Def. 9, we get $t \equiv^{\alpha}_0 t'$.
$\Leftarrow|$ We will see that $\mathcal{R} = \{t,t' \in FyTrees(\mathbb{A}) \mid t \equiv^{\alpha}_0 t'\}$ is a bisimulation. We need to check the bisimulation conditions for any $(t,t') \in \mathcal{R}$. If we obtain $t \equiv^{\alpha}_0 t'$ by application of the first clause in Def. 8, we immediately have $t \sim t'$, since bisimilarity is a congruence. We can only apply the second clause in Def. 8 to derive $t \equiv^{\alpha}_0 t'$ in the trivial case in which $b = a$, so that we would have $t' = t$, and then $t \sim t'$. Finally, if $t = (\sum_{j\in J} a_j t_j) + at''$, $t' = (\sum_{j\in J} a_j t_j) + at'''$, and $(t'',t''',0) \in \mathcal{D}$ enables us to obtain $t \equiv^{\mathcal{D},\alpha}_0 t'$, then the only non trivial cases correspond to the a-derivatives of these trees, and then the result is immediate, since $(t'',t''',0) \in \mathcal{D}$.

The second part is a consequence of Prop. 3. □

Lemma 1 Any sequence $\mathcal{S}$ producing $t \rightsquigarrow_{\alpha,d} t'$ can be "factorized" into an "structured" sequence $\mathcal{T} := t = t^{0,2} \rightsquigarrow_{\alpha,d_{11}} t^{1,1} \rightsquigarrow^1_{\alpha,d_{12}} t^{1,2} \rightsquigarrow_{\alpha,d_{21}} \cdots \rightsquigarrow^1_{\alpha,d_{k2}} t^{k,2} \rightsquigarrow_{\alpha,d_{(k+1)1}} t^{k+1,1} = t'$, where $\sum d_{1i} + \sum d_{2i} = d$, and the distance steps $t^{l,1} \rightsquigarrow^1_{\alpha,d_{l2}} t^{l,2}$ in it are exactly all the first level steps in $\mathcal{S}$. So that, no one of the subsequences producing $t^{l,2} \rightsquigarrow_{\alpha,d_{(l+1)1}} t^{l+1,1}$ contains any first level step.
Proof. Immediate. By the way, the only goal of this lemma is to introduce the notation for the intermediate trees that correspond to the first level steps in the sequence. □

Proposition 6 Any sequence $\mathcal{S}^l$ producing $t^{l,2} = \sum_{i=1}^m a_i t_i \rightsquigarrow_{\alpha,d_{(l+1)1}} t^{l+1,1} = \sum_{i=1}^m a_i t'_i$ can be reordered getting an "ordered" sequence $\mathcal{O} := t^{l,2} = \sum_{i=1}^m a_i t_i \rightsquigarrow_{\alpha,d^1_{(l+1)1}} \sum a_i t^1_i \rightsquigarrow_{\alpha,d^2_{(l+1)1}} \sum a_i t^2_i \rightsquigarrow_{\alpha,d^3_{(l+1)1}} \cdots \rightsquigarrow_{\alpha,d^m_{(l+1)1}} \sum a_i t^m_i = \sum_{i=1}^m a_i t'_i = t^{l+1,1}$, where $\sum_{j=1}^m d^j_{(l+1)1} = d$, $t^j_i = t'_i \ \forall j \leq i$, and $t^j_i = t_i \ \forall j > i$.

This means that for each $j \in \{1, \ldots, m\}$ $\sum a_i t^{j-1}_i \rightsquigarrow_{\alpha,d^j_{(l+1)1}} \sum a_i t^j_i$ corresponds to $a_j t_j \rightsquigarrow_{\alpha,d^j_{(l+1)1}} a_j t'_j$, so that the distance steps in the former are exactly those from $\mathcal{S}^l$ working at the corresponding summand $a_j t_j$ of t. As a consequence, for each $j \in \{1, \ldots, m\}$ we also have $t_j \rightsquigarrow_{\alpha,(d^j_{(l+1)1})/\alpha} t'_j$, which is obtained by removing the common prefix a_j from the steps of the subsequence generating $a_j t_j \rightsquigarrow_{\alpha,d^j_{(l+1)1}} a_j t'_j$.
Proof. The intermediate sequences $\mathcal{S}^l$ do not correspond to first level steps. Therefore, the first level structure of all the trees along it has to be the same (thus, that of both $t^{l,2}$ and $t^{l+1,1}$). Moreover, if $t^{l,2} = \sum_{i=1}^n a_i t_i$ and $t^{l+1,1} = \sum_{i=1}^m a_i t'_i$, any of the steps in $\mathcal{S}^l$ participates in the rewriting of some t_i into the corresponding t'_i. Clearly, when corresponding to different indexes these steps can be reordered as preferred: we will reorder them so that the indexes to which they correspond will form a non-decreasing sequence and this will produce he desired decomposition of $\mathcal{S}^l$, whose components $\sum a_i t^{j-1}_i \rightsquigarrow \sum a_i t^j_i$ correspond to the rewritings of t_j into t'_j, taking $t^0_i = t_i$. □

Proposition 8 For all $d \in \mathbb{R}^+$, $\alpha \in (0, 1]$, the relations $\equiv^\alpha_d$ and $\widehat{\equiv}^\alpha_d$ are equal.
Proof. $\overline{\Rightarrow|}$ Since rules 1 and 2 in Def. 8 are clearly particular cases of rule 1 in Def. 11, we would immediately have $t \equiv^\alpha_d t' \Rightarrow t \widehat{\equiv}^\alpha_d t'$.
$\overline{\Leftarrow|}$ For the converse implication we have to see that any family of relations $\mathcal{D}' = \{\widehat{\equiv}^{\mathcal{D}\alpha}_d\}$ in an α-ccd. We proceed by rule induction o the generation of the pairs in this relation:

- If $t \widehat{\equiv}_d^{\mathcal{D}\alpha} t'$ is generated by rule 1 in Def. 11, we have $t \rightsquigarrow^1_{\alpha,d} t'$. If this is a first level step, then it corresponds to the application of either rule 1 or rule 2 in Def. 8 showing the result.
- Otherwise, we simply proceed by induction o the level to which this step corresponds. If it is a k-level step with $k > 1$ then we have $t = \sum_{i\in I} a_i t_i$ and $t' = \sum_{i\in I} a_i t'_i$ where there exists some $j \in J$ such that $t'_i = t_i \ \forall i \neq j$ and $t_j \rightsquigarrow^1_{\alpha,\frac{d}{\alpha}} t'_j$ is a $(k-1)$-level step. Now, $t_j \widehat{\equiv}_{\frac{d}{\alpha}}^{\mathcal{D}\alpha} t'_j$ by i.h. and therefore, the coinductive transformation sequence $\mathcal{C} := t = \sum a_i t_i \widehat{\equiv}_d^{\mathcal{D}\alpha} \sum a_i t'_i = t'$ confirms that the conditions in Def. 9 are satisfied in this case.
- If $t \widehat{\equiv}_d^{\mathcal{D}\alpha} t'$ is generated by rule 2, we have $t = t^1 + at^2$, $t' = t^1 + at^3$ with $(t^2, t^3, \frac{d}{\alpha}) \in \mathcal{D}$. Now, it is enough to prove that $(t, t', \mathcal{D})$ implies $t \widehat{\equiv}_d^{\mathcal{D}\alpha} t'$. We obtain $(t, t', d) \in \mathcal{D}$ form a transformation sequence $\mathcal{C}$ which uses first steps and pairs in $\mathcal{D}$ substituted at a deeper level. Now, we proceed by an immediate induction on the length of the sequence, studying the two possible kinds of steps in it:
  - For the first level steps it was already proved above.
  - If we have applied a substitution in $\mathcal{D}$ at a deeper level, we have again $t = t^1 + at^2$, $t' = t^1 + at^3$ with $(t^2, t^3, \frac{d'}{\alpha}) \in \mathcal{D}$, and therefore $t \widehat{\equiv}_{d'}^{\mathcal{D}\alpha} t'$.

□

8.6 For a further formalization of some results on trees and bisimulation

Considering trees as representatives of the semantics of processes, we noticed that some care should be taken when specifying the structures in which we are working on. We dived into the literature searching where the notation and formal definitions of that structures have been stablished. Unfortunately, we were unable to find anything satisfying our expectations. Hence, we decided to write by ourselves the material that we present now in this Section.

The correctness of our results in the published papers was implicitly supported by the following material. We avoided to include it in the original version of each of these papers in order to facilitate the comprehension of our results and proofs. The reason to incorporate the material here is to provide the interested reader with the full details for operating with trees. For instance, this would be the required case to check our results by means of any proof assistant such as *Isabelle* or *Coq*.

8.6.1 On classes of labeled trees

Let us start by recalling the coalgebraic definition of labeled transition systems (lts).

Definition 1 Labelled Transition Systems (lts) on a set of actions $\mathbb{A}$, are coalgebras $succ : N \to LTS(N, \mathbb{A})$ of the functor $LTS(N, \mathbb{A}) = \mathcal{P}(\mathbb{A} \times N)$. We say that the (fixed) set $\mathbb{A}$ is the alphabet of actions of the system, and N its set of states. A lts with initial state is just a lts $(N, succ)$ where some distinguished (initial) state $n_0 \in N$ is fixed.

In the following we will usually denote a lts by its set of states, leaving implicit the corresponding *succ* function. As usual, we say that $n \xrightarrow{a} n'$ is a transition of a system $(N, succ)$ when $(a, n') \in succ(n)$. Given $\overline{a} \in \mathbb{A}^*$ with $\overline{a} = a_1 \dots a_k$, we have a computation $n \stackrel{\overline{a}}{\Longrightarrow} n'$ if and only if there exists a sequence $n \xrightarrow{a_1} n_1 \xrightarrow{a_2} n_2 \cdots \xrightarrow{a_k} n_k = n'$. Taking $n = n_0$ we obtain the computations of the lts with initial state (N, n_0), and we say that the sequence $n_0 a_1 n_1 \dots a_k n_k$ with $(a_{i+1}, n_{i+1}) \in succ(n_i)\ \forall i \in \{0 \dots k-1\}$ is a path in (N, n_0). We denote the set of paths as $Path(N, n_0)$.

We say that the system N has *finite-state* when $|N| < \infty$; we say that the system (N, n_0) is *finite*, when it only admits a *finite set of computations*. Finally, we say that (N, n_0) has only *finite computations* when does not exist infinite computation $n_0 \xrightarrow{a_1} n_1 \xrightarrow{a_2} n_2 \dots$. We say that a system N is *finitely branching*, when for all $n \in N$ we have $|succ(n)| < \infty$. We can directly define the class of finitary trees by changing the $\mathcal{P}$ operator that appears in Def. 1 by the finite parts operator $\mathcal{P}_f$.

When we study lts's with an initial state (N, n_0), we will usually assume that all the states in N are reachable from n_0, that is $\forall n \in N\ \exists \overline{a} \in \mathbb{A}^*$ with $n_0 \stackrel{\overline{a}}{\Longrightarrow} n$. Next we present some representative examples that illustrate the different classes of systems that we have defined above. See Fig. 8.1 for a pictorial representation.

Example 1 We can get a single degenerated empty lts taking $N = \emptyset$, but in order to have a lts with initial state we at least need $n_0 \in N$ for some n_0. Then, the simplest such system is that defined by $N_0 = \{n_0\}$, with $succ(n_0) = \emptyset$. Other interesting systems are those with a single transition: $N_1 = \{n_0, n_1\}$, $succ(n_0) = \{(a, n_1)\}$, with $a \in \mathbb{A}$ and $succ(n_1) = \emptyset$.

Example 2 Two simple finite-state systems that have infinitely many computations are the following: $N_{1,\infty} = n_0$, with $succ(n_0) = \{(a, n_0)\}$; $N_{2,\infty} = \{n_0, n_1\}$, with $succ(n_0) = succ(n_1) = \{(a, n_0), (a, n_1)\}$.

Example 3 Another simple interesting finite-state system is that defined by $N'_{2,\infty} = \{n_0, n_1\}$, but taking $succ(n_0) = \{(a, n_0), (a, n_1)\}$ and $succ(n_1) = \emptyset$.

Example 4 Next we present three interesting non-finitely branching systems:

1. $N_{\mathbb{N}} = \mathbb{N}$ taking also $\mathbb{A} = \mathbb{N}$, and $n_0 = 0$ with $succ(0) = \{(k, k) \mid k \in \mathbb{N}\}$, $succ(k) = \emptyset$, $\forall k \geq 0$.
2. $N_2 = \{0\} \cup \{(i, j) \mid i, j \in \mathbb{N}, 1 \leq j \leq i\}$, taking $n_0 = 0$ with $succ(0) = \{(a, (n, 1)) \mid n \in \mathbb{N}\}$ and $succ((i, j)) = \{(a, (i, j + 1))\}$ when $j < i$, while $succ((i, i)) = \emptyset$.
3. $N_2^+ = N_2 \cup \{(\infty, n) \mid n \in \mathbb{N}\}$, changing also the definition of $succ$, taking $succ(0) = \{(a, (x, 1)) \mid x \in \mathbb{N} \cup \{\infty\}\}$ and $succ((\infty, j)) = \{(a, (\infty, j + 1))\}$.

We can define (rooted) trees as a particular class of lts's:

Definition 2 We say that a system (N, n_0) is (or defines) a *tree* iff for all $n \in N$ there is a single path $n_0 a_1 n_1 \dots a_k n_k$ with $n_k = n$. Then, we say that each node n_k is at *level* k, and define $Level_k(N) = \{n \in N \mid n \text{ is at level } k \text{ in } N\}$. We define the *depth* of a tree N as $depth(N) = max\{l \in \mathbb{N} \mid Level_l(N) \neq \emptyset\}$.

We denote by $Trees(\mathbb{A})$ the set of trees on the set of actions $\mathbb{A}$. In the following, we will usually use t, t_1, t_i, ... to denote trees, leaving implicit their sets of nodes N, and we will say that the initial node n_0 is the *root* of the tree. We define the families of *trees with bounded depth* $FDTrees_k(\mathbb{A})$, by $FDTrees_k(\mathbb{A}) = \{t \in Trees(\mathbb{A}) \mid depth(t) \leq k\}$. Moreover, we denote by $FyTrees(\mathbb{A})$ the collection of *finitary trees* in $Trees(\mathbb{A})$. For any tree $t \in Trees(\mathbb{A})$ we define its *first-level width*, that we will represent by $||t||$, as $||t|| = |Level_1(t)|$. We also define the *first k-levels width of t*, denoted by $||t||_k$, as $||t||_k = max\{||t_n|| \mid n \in \bigcup_{l \leq k} Level_l(t)\}$.

All the classes of lts's introduced above can also be defined for trees, but in this case several of them become the same. In particular, the famous *König's lemma* asserts the following:

Proposition 1 The finitary trees with no infinite computations are exactly the finite trees.

All the systems in Ex. 1 and Ex. 4, but the empty system $N = \emptyset$, are indeed trees. In particular, we will usually denote by $\mathbf{0}$ the empty tree N_0. Instead, those in Ex. 2 and Ex. 3 are not trees. However, any lts with an initial state $(N, succ, n_0)$ induces a tree of computations in a canonical way, by means of the notion of *unfolding*.

Definition 3 Given a lts with initial state $(N, succ, n_0)$, we define its unfolding $unfold(N)$ as the tree $(\overline{N}, \overline{succ}, \overline{n_0})$, where $\overline{N} = Path(N, n_0)$, $\overline{succ}(n_0 a_1 \dots n_k) =$

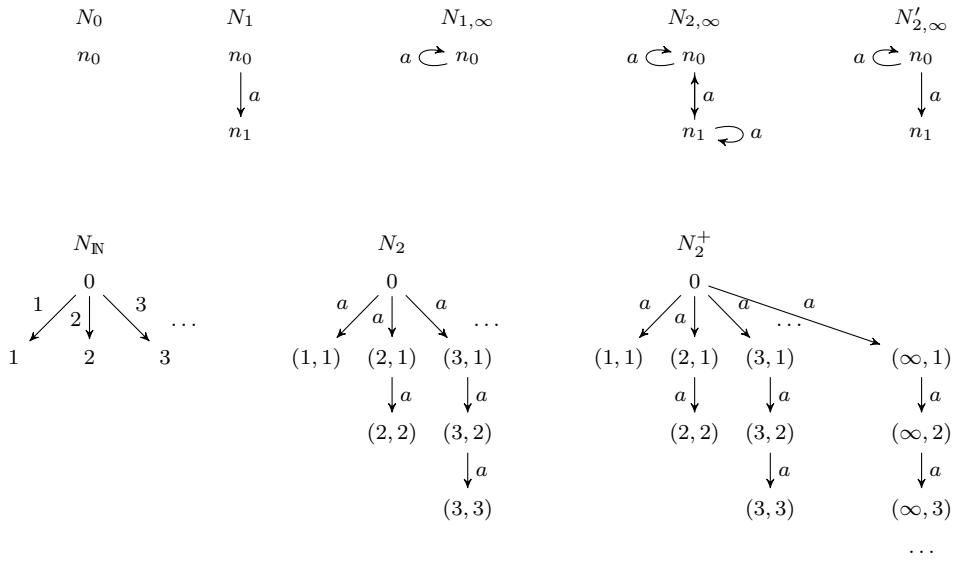


Figure 8.1: Labelled Transitions Systems and Trees used in Examples 1-4

$\{(a, n_0 a_1 \dots n_k a n') \mid (a, n') \in succ(n_k)\}$, and $\overline{n_0} = n_0$.

Proposition 2 Any unfolding of a lts is indeed a tree.

As a consequence, in the following, by abuse of notation, we will sometimes define a tree by means of a description of the lts which generates it via unfolding.

Example 5 The lts $N_{1,\infty}$ in Ex. 2 generates the infinite-list tree with arcs labeled by a all along the list; while $N_{2,\infty}$, generates the complete binary tree labeled in the same way.

The trees that can be generated by unfolding a finite state system are called rational in the literature [Gue81]. But when we use trees to describe the semantics of processes, we are not (usually) interested at all in the identity of the nodes that constitute the tree, but only on the structure defined by its branches. We need some mechanism to abstract away those identities. This can be done by means of *isomorphisms* on trees.

Definition 4 Given two trees $(N_1, succ_1, n_0^1)$ and $(N_2, succ_2, n_0^2)$, we say that a bijection $i : N_1 \to N_2$ is an isomorphism between these two trees if $i(n_0^1) = n_0^2$, and for all $n \in N_1$ $succ_2(i(n)) = \{(a, i(n') \mid (a, n') \in succ_1(n)\}$. When this is the case, we write $N_1 \equiv N_2$. We call *abstract trees* to the equivalence classes induced by the equivalence relation defined by bijections between trees.

As usual, we denote the abstract tree represented by a tree $(N, succ, n_0)$ by $[(N, succ, n_0)]$. The class of abstract trees over the set of actions $\mathbb{A}$ is denoted by $ATrees(\mathbb{A})$. We also use names with an initial A for all the other particular classes of trees that we have defined above, when defined on abstract trees: $AFTrees(\mathbb{A})$, $AFyTrees(\mathbb{A})$, ...

It is obvious that whenever we have a tree $t = (N, succ, n_0)$ each node $n \in N$ induces a tree $t_n = (N_n, succ, n)$. We say that all these trees are the *subtrees* of t. In particular, this is the case for the set of *children* of the root, which are those nodes in $Level_1(N, succ, n_0) = \{n_{1,j} \mid \exists\, (a_j, n_{1j}) \in succ(n_0)\}$. We will decompose any tree t into the formal sum $\sum_{n_{1j} \in Level_1(t)} a_j t_{n_{1j}}$. In particular, when $|Level_1(t)| = 1$, we have $t = t'$, which can be reversed to define the tree at' starting from $a \in \mathbb{A}$ and $t' \in ATrees(\mathbb{A})$. In a similar way, if $Level_1(t) = N_1 \cup N_2$ is a disjoint decomposition of that set, we can write $t = \sum_{n_{1j} \in N_1} a_j t_{n_{1j}} + \sum_{n_{1k} \in N_2} a_k t_{n_{1k}}$, that we can also reverse to define the sum $(+)$ of abstract trees.

By applying the decomposition above we can "recover" the tree-structure of any abstract tree.

Definition 5 Each abstract tree $[(N, succ, n_0)]$ induces a canonical tree given by the decomposition $[(N, succ, n_0)] = \sum_{n_{1j} \in Level_1(N, succ, n_0)} a_j [N_{n_{1j}}]$.

Remark It is clear that in order to get a tree, the sets of nodes of the main subtrees, $N_{n_{1j}}$ of a tree have to be pairwise disjoint. Instead, when generating our canonical representatives of the abstract trees as above, we can have several subtrees N_n that are isomorphic, but when "introducing" the corresponding $[N_n]$ nodes in the definition of the canonical tree $[(N, succ, n_0)]$ we must take care of that using "separate copies" of those subtrees in order to preserve the tree structure of the whole.

Example 6 If we consider the tree $(\{n_0, n_1, n_2\}, succ, n_0)$ with $succ(n_0) = \{(a, n_1), (a, n_2)\}$ and $succ(n_1) = succ(n_2) = \emptyset$, its corresponding abstract tree has still three nodes, even if N_{n_1} and N_{n_2} are isomorphic. Of course, this must be the case in order to have the expected property:

> The canonical tree attached to any abstract tree is indeed a representative of the corresponding isomorphic class.

Usually, we graphically represent abstract trees using different but no named dots, to denote their nodes. But when we want to describe by means of an abstract tree the set of computations (behaviors) of a system, it will be probably the case that we will not be interested on distinguishing pairs of trees as N_{n_1} and N_{n_2}.

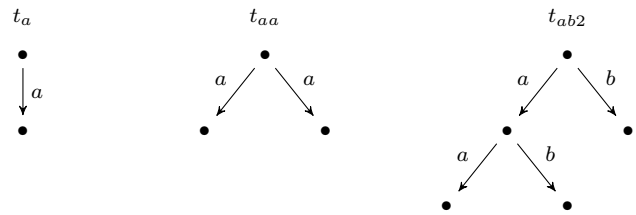


Figure 8.2: Graphical representation of abstract Trees

This certainly needs some additional elaboration, since definitely they are not isomorphic.

Definition 6 We say that a tree $(N, succ, n_0)$ is *pure* when for all $n \in N$, whenever we have $(a, n'), (a, n'') \in succ(n)$ and $n' \neq n''$, we also have $n' \not\equiv n''$.

Example 7 The tree t_a in Fig. 8.2 is pure, but t_{aa} contains two equivalent subtrees under its root. We allow several subtrees of a pure tree to be isomorphic, as far as they do not appear as children of the same node and connected to it by the same action. We have, indeed, that t_{ab2} in the figure is pure.

As a consequence of the decomposition property, pure abstract trees can be represented by *a set* of pairs (a_i, t_i), where $a_i \in \mathbb{A}$ and the t_i's are also pure. Instead, arbitrary abstract trees correspond to *multisets* of those pairs, instead of plain sets.

Proposition 3 Any abstract tree t has an associated pure tree $Pure(t)$ that is obtained by removing any repeated child of any node, leaving a single representative for each class.

Example 8 Tree t_a in Fig. 8.2 is the pure form of t_{aa}. A more interesting case appears in Ex. 5, where the tree $unfold(N_{1,\infty})$ is the reduced form of the tree $unfold(N_{2,\infty})$.

Bisimulation has been introduced as an alternative way to capture the equivalence of trees in a natural (coinductive) manner. Next we recall its formal definition.

Definition 7 Let $\mathcal{R}$ be a relation on $Trees(\mathbb{A})$. We say that $\mathcal{R}$ is a bisimulation when for all $(t, t') \in \mathcal{R}$ we have

- $\forall\ (a, t_1) \in succ(t)\ \exists\ (a, t'_1) \in succ(t'),\ (t_1, t'_1) \in \mathcal{R}$.
- $\forall\ (a, t_2) \in succ(t')\ \exists\ (a, t'_2) \in succ(t),\ (t'_2, t_2) \in \mathcal{R}$.

We say that t and t' are bisimilar when there exists some bisimulation $\mathcal{R}$ such that $(t, t') \in \mathcal{R}$, and then we write $t \sim t'$.

Next we only recall some important properties of bisimulations.

Proposition 4 Bisimilarity on finite pure trees is simply isomorphism:

$$\forall t, t' \in FTrees(\mathbb{A})\ t \sim t' \Leftrightarrow t \equiv t' \Leftrightarrow [t] = [t'].$$

Remark This corresponds to the fact that the set of axioms ACI

- $(p_1 + p_2) + p_3 \sim p_1 + (p_2 + p_3)$ (associativity),
- $p_1 + p_2 \sim p_2 + p_1$ (commutativity), and
- $p \sim p + p$ (idempotency)

constitutes a sound and complete axiomatization of bisimulation equivalence on finite processes, when the binary choice operator + is used to generate the branched structure of the processes [Mil89]. When we use the tree notation for abstract trees, we are implicitly using associativity and commutativity, so that only the idempotency law is needed; it empowers us by adding, or remove, finitely many copies of the same abstract tree below each node of a tree, obtaining (finite) trees that are always bisimilar (and isomorphic) to that tree.

But, bisimilarity is coarser than isomorphism, when we consider arbitrary (possibly infinite) trees:

Example 9 It is easy to check that the tree $unfold(N'_{2,\infty})$, with $N'_{2,\infty}$ as in Ex. 3, is pure. However, for the system $N_{1,\infty}$ in Ex. 2 we have $unfold(N'_{2,\infty}) \sim unfold(N_{1,\infty})$, but, obviously, $unfold(N'_{2,\infty}) \not\equiv unfold(N_{1,\infty})$, even if both are pure.

It is by means of this kind of examples that bisimulation, and in general the conductive methods, show their (specific) interest: they provide a simple way to define natural equivalences between systems (or equivalently trees) when an inductive approach would require a much more complicated procedure using some (elaborated) notions of finite approximations of trees by finite trees, and the adequate notion of continuity.

Definition 8 Given a tree $t = (N, succ, n_0)$ and a natural number $k \in \mathbb{N}$, we define its k-th cut, $\pi_k(t)$, as the restriction of t to the nodes in $\bigcup_{l \leq k} Level_l(N)$:

$\pi_k(t) = (\pi_k(N), succ_k, n_0)$, where $\pi_k(N) = \cup_{l \leq k} Level_l(N)$, $succ_k(n) = succ(n)$ when $n \in \cup_{l<k} Level_l(N)$, and $succ_k(n) = \emptyset$ when $n \in Level_k(N)$.

We have denoted these projections by π_k because whenever they are applied to finitary trees, they constitute a projective sequence [Plo76] whose limit is the original tree. We can see the tree $\pi_k(t)$ as the (natural) projection of t on the set $FDTrees_k(\mathbb{A})$.

Proposition 5 For any $t \in Tree(\mathbb{A})$ and $l, k \in \mathbb{N}$ with $l \leq k$, we have $\pi_l(\pi_k(t)) = \pi_l(t)$. Any finitary abstract tree is unequivocally defined by its sequence of projections: $\forall t, t' \in FyTrees(\mathbb{A})$ $(\forall k \in \mathbb{N}\ \pi_k(t) \sim \pi_k(t')) \Rightarrow t = t'$.

As a matter of fact, whenever $l \leq k$ we have indeed $\pi_l(N) \subseteq \pi_k(N)$, and then we can see each $\pi_l(t)$ naturally embedded into $\pi_k(t)$. Even more, we can obtain this last projection by "expanding" the nodes of $\pi_l(t)$ at level l, adding the corresponding subtrees of $\pi_k(t)$ that have those nodes as roots. In this way, one can see t as a "telescopic" expansion, where more and more levels are "opened" by expanding the leaves of each $\pi_l(t)$ to obtain $\pi_{l+1}(t)$, and so on. This is why we like to call "telescopic" to the projective sequences $\{\pi_k(t) \mid k \in \mathbb{N}\}$.

Example 10 The result above becomes false when we consider infinitary trees. For the trees N_2 and N_2^+ in Ex. 4 we have $\pi_k(N_2) \equiv \pi_k(N_2^+)$ $\forall k \in \mathbb{N}$, since the "additional" branch executing a^k provided by N_2^+ can be "absorbed" by the infinitely many such branches that we already have in $\pi_k(N_2)$. As a matter of fact, since we also have $t \not\sim t'$, this is also a counterexample disproving the continuity of bisimilarity wrt the approximations provided by the projections π_k, when we allow infinitary trees.

Instead, when we restrict ourselves to finitary trees, we have the following.

Proposition 6 For all trees $t, t' \in FyTrees(\mathbb{A})$ we have $(\forall k \in \mathbb{N}\ \pi_k(t) \sim \pi_k(t'))$ $\Rightarrow t \sim t'$.

It is interesting to observe that, even for finitary trees, isomorphism between pure trees is not continuous in general.

Example 11 Let us recall that the trees $unfold(N_{1,\infty})$ and $unfold(N'_{2,\infty})$ in Ex. 9 satisfied $unfold(N'_{2,\infty}) \sim unfold(N_{1,\infty})$, but $unfold(N'_{2,\infty}) \not\equiv unfold(N_{1,\infty})$. Instead, it is easy to check that $\forall k \in \mathbb{N}$ we have $Pure(\pi_k(unfold(N'_{2,\infty}))) \equiv Pure(\pi_k(unfold(N_{1,\infty})))$.

If we want to remain in the universe of pure trees, then when cutting a tree we need to apply the *Pure* operator, since two no isomorphic subtrees may turn

Figure 8.3: Two ordered trees

isomorphic when cut. Therefore, we need to remove possible duplications in order to obtain the corresponding pure projections.

The result above justifies the adoption of bisimilarity as the natural way to define the equivalence between (finitary) pure abstract trees. In fact, it can be proved that bisimilarity is the finest continuous equivalence relation coarser than isomorphism, for this class of abstract trees. Therefore, we can say that bisimilarity is the continuous closure of isomorphism wrt the approximations of finitary trees by their finite projections.

8.6.2 Ordered Trees as representations of Finitary Abstract Trees

Ordered trees are a quite convenient way to represent trees in a more clear way especially when we are working with finitary trees.

Definition 9 Finitary *ordered trees* are defined by replacing the operator $\mathcal{P}_f$ that appears in the definition of finitary trees, by its "ordered" version $\bigcup_{k\in\mathbb{N}}(\cdot)^k$. The class of finitary ordered trees on $\mathbb{A}$ will be denoted by $OFyTrees(\mathbb{A})$.

For these *ordered trees* the order between the children of each node becomes important, which is equivalent to remove the commutative law from the axiomatization for finite trees. It is clear that we have a *forget_ord* application between ordered trees and trees, that is obtained from the *forget_ord* function from $\bigcup_{k\in\mathbb{N}} X^k$ into $\mathcal{P}_f(X)$, that corresponds to finite enumerations of finite subsets of X. We abstract away these enumerations when working with finite sets, in particular when we consider textual or graphical representations of these sets (e.g. $\{1,2\} = \{2,1\}$). We will do exactly the same for ordered and plain finitary trees.

Obviously, the two trees in Fig. 8.3 are not equal as ordered trees, but both are representatives of the same (plain) abstract tree. In fact, the class of finitary abstract trees can be obtained as a quotient from the class of finitary ordered trees wrt the equivalence relation induced by the simultaneous permutation of the sequences of children of all the nodes in a tree. Then, those ordered trees become the representatives of the classes defining plain finitary abstract trees.

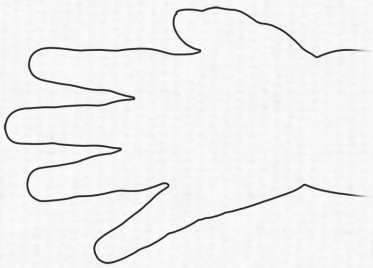
"...which is why people say: that each man is the architect of his own fortune."

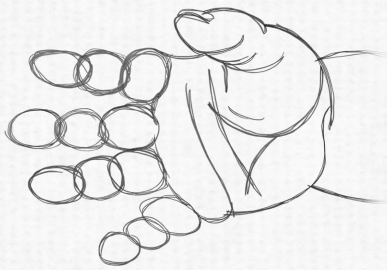
"...y de aquí viene lo que suele decirse: que cada uno es artífice de su ventura."

MÁS SOBRE EQUIVALENCIAS LÓGICAS Y DISTANCIAS ENTRE PROCESOS
Revisiting logical semantics for processes and their distances

DAVID ROMERO HERNÁNDEZ
Tesis Doctoral | Madrid, 2016 | Director: David de Frutos Escrig

Departamento de Sistemas Informáticos y Computación
Facultad de Informática | Universidad Complutense de Madrid





"... and it is necessary for us to use our knowledge and discernment to distinguish between these two kinds of knights, so similar in names, so dissimilar in actions."

El Ingenioso Hidalgo Don Quijote de La Mancha | Miguel de Cervantes Saavedra | Chapter·3 Part·2

"... y es menester aprovecharnos del conocimiento discreto para distinguir estas dos maneras de caballeros, tan parecidos en los nombres y tan distantes en las acciones."